Performance Modeling of a Computer System Saudi Arabian Ministry of Interior Online System

by

Khaled W. Al-Dhaher

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

COMPUTER SCIENCE

June, 1987

Order Number 1355711

Performance modeling of a computer system: Saudi Arabian Ministry of Interior online system

Al-Dhaher, Khaled W., M.S.

King Fahd University of Petroleum and Minerals (Saudi Arabia), 1987

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

PERFORMANCE MODELING OF A COMPUTER SYSTEM SAUDI ARABIAN MINISTRY OF INTERIOR ONLINE SYSTEM

BY

Khaled W. AL-Dhaher

A Thesis Presented to the
FACULTY OF THE COLLEGE OF GRADUATE STUDIES
KING FAHD
UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
IN
COMPUTER SCIENCE

June, 1987

PRINTED BY: UPM PRESS

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

DHAHRAN, SAUDI ARABIA

THE GRADUATE SCHOOL

This thesis, written by

Khaled Walid AL-Dhaher

under the direction of his Thesis Committee, and approved by all its members, has been presented to and accepted by the dean of the Graduate School, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

Abdulaziz Al-Zaki

Dean of the Graduate School

Date : May 11, 1987

Department Chairman

Thesis Committee

M. Uşşak Gözlayan

Chairman

Member

Member

This Thesis is dedicated to my Father, Mother , Brothers and Sister for their continuous encouragement and support.

ACKNOWLEDGMENTS

Acknowledgment is due to the King Fahd University of Petroleum and Minerals, the National Information Center of the Saudi Arabian Ministry of Interior, and the Saudi Arabian National Center for Science and Technology, that supported this research under project number NP:SAMIS.

I wish to express my appreciation to Dr. Mehmet U. Caglayan, who served as my thesis advisor. I also wish to thank the other members of my Thesis Committee, Dr. Mohammad I. AL-Suwaiyel, and Dr. Zafarullah Zafar.

TABLE OF CONTENTS

List of Figures

List of Tables

الخلاصة

تهدف الدراسة إلى تطوير طرق لبناء نموذج محاكاة لنظام الحاسب الآلي في مركز المعلومات الوطني التابع لوزارة الداخلية في المملكة العربية السعودية. وتعتبر هذه الدراسة جزءاً من مشروع متكامل بتمويل من مركز المعلومات الوطنى لدراسة أسباب الأداء غير المرضي للنظام.

هذا وقد تم بناء النموذج الجوهري للنظام و بناء أحد نماذج الأنظمة التطبيقية التابعة له باستخدام طريقة بناء النماذج المقترحة. ومن ثم تم اختبار النموذج المتكامل ورصد السلوك الملاحظ تحت ظروف مختلفة ، ومن ذلك يمكن تصوير إتجاه الأداء الوظيفي مما يؤدي إلى فهم أفضل لرد فعل النظام تحت هذه الظروف.

إن الملاحظة الأساسية لهذه الدراسة هي الإستخدام غير المتكافىء لبعض الأجهزة التي تسبب إختناقات في النظام بحيث تعطل مرور المعلومات ، مما يؤثر على الأداء العام للنظام . وقد تم إثبات وجود مثل هذه الإختناقات المرورية عن طريق إجراء التجارب اللازمة على النموذج.

ABSTRACT

The main objective of this thesis is to establish a modeling methodology that enables the development of a computer simulation model of the SAMIS Online system. The modeled system is operated by the National Information Center (NIC), Riyadh. This thesis work is part of a project supported by NIC and has been initiated due to the unsatisfactory performance of the system.

A core model has been constructed, and the suggested modeling methodology has been utilized to construct a model of one of the SAMIS Online System Application Subsystems. The combined model has been tested under different configurations, and its behaviour has been documented. The observed behaviour of the model enables the construction of performance trends leading to a better understanding of the reaction of the SAMIS system under different situations.

The main observation of this study was the over and under utilization of different hardware components of the SAMIS Online System, resulting in bottlenecks that affected the performance of the system as a whole.

1. INTRODUCTION

The primary purpose of this work was to study the performance of the Saudi Arabian Ministry of Interior System (SAMIS), more specifically the Online System, which is one of two major components of SAMIS, the other being the Administrative System.

The work was initiated as part of a project funded by the Saudi Arabian National Center for Science and Technology (SANCST) for an initial duration of eighteen months, on behalf of the National Information Center, Saudi Arabian Ministry of Interior. The reason for proposing the work was due to the poor response time observed by the users of the SAMIS Online System.

The SAMIS Online System is a computer based information system operated by the National Information Center (NIC), Ministry of Interior (MOI) Riyadh, to process vital information about the citizens and expatriates in the Kingdom of Saudi Arabia.

The SAMIS Online System has more than one thousand terminals connected to it in about two hundred and fifty sites distributed in the Kingdom of Saudi Arabia. It supports three major components, namely the general services component responsible for keeping records of civil registration, drivers licences, vehicle

registration, and pilgrim services. The second component which is the security component is responsible for keeping records of the passport issuance, miscreant, criminal records, alien control and border control. The third component is a support system responsible for storing and retreiving micrographics systems, and for message communication. Thus, the Online System supports actually eleven subsystems.

The SAMIS system was studied before the work on the development of modeling methodology and the construction of the model itself have been initiated. Following the proposed methodology, a computer simulation model was built to study the performance of the SAMIS Online System. The study of the SAMIS Online System included the details of the hardware, software and application system components.

The methodology has been developed and tested on one of the subsystems. Then, the proposed methodology has been applied to construct a main model together with the model of one subsystem. Following the same proposed modeling methodology, construction of other subsystems could be incorporated in the main model as future extensions of this work. The addition could be done by following the abstraction process of the software system component, described in this thesis.

A computer simulation model to evaluate the performance

of the SAMIS Online System has been built and tested. The model was developed by using a specialized computer simulation language called the Performance Analyst's Workbench System (PAWS), distributed by the Information Research Associates (IRA) Austin, Texas.

The thesis text is going to be presented in the following manner. In chapter two, an overview of performance evaluation will be presented to describe different measurement and modeling techniques. This chapter will focus on the modeling techniques, and formulation and construction of simulation models.

In chapter three, an overview of the Performance Analyst's Workbench System (PAWS) will be presented to describe the features of the language and the process of model development while using this language. An example of a simple system with the proper modeling construction steps using PAWS would then be presented.

In chapter four, an overview of the SAMIS system will be presented focusing on the Online System. Details of the hardware and software components that are of importance for the study of the SAMIS Online System are given. Then the SAMIS Application System and details of the method to interface it with the SAMIS system elements such as the operating system and the database management system are presented.

In chapter five, a discussion of the methodology of abstraction of the SAMIS Online System is presented, and the methodology used to build the model is discussed. The construction and the details of the Information Processing Graph (IPG) of the SAMIS Online System model are described. Then, the details of the simulation model built as a PAWS program are presented.

In chapter six the experimentation process and procedures conducted on the model are discussed, including reasoning for the choice of manipulation of parameters, and test values, and the explanation of the features of interest that were expected and sought. Then the results of the experimentation were analyzed and the formulated opinion, recommendations, and conclusion of the work are presented.

The main observation made by the study of experiment results conducted on the model show the over and under utilization of some hardware components resulting in system bottlenecks leading to the observed poor performance.

2. PERFORMANCE EVALUATION

2.1. Introduction

Engineering systems in general are evaluated by all people in direct contact with them, ranging from the designer to the user, each of which looks at the system from a different point of view.

The study of the performance of computing systems involves the study of all the components of these systems, including Computer Networks, Operating Systems, Database Management Systems, and Application Programs. Each of these subsystems can seperately be studied to determine its performance. The performance of the overall computer system, composed of all of these systems, should also be studied as a whole, due to the fact that the behaviour of a system in isolation does not reflect its behaviour in relation with other systems.

What is evaluated is not only what the machine is doing, or is able to do what is expected from it, but also how it does it and at what cost. Cost does not always mean direct financial effect, it might mean time, memory utilization, effect on other interacting systems, among other things. All of these factors should be taken into account when the performance of a specific system is evaluated. If the performance of that system was found

not to be satisfactory, then something should be done to improve it.

The action taken actually depend on the state of the system. If the system studied is a proposed system, and is still under design, then it is feasible to improve the design, or change the selected setup to overcome the performance problem. On the other hand, if the system already exists, then improving its performance by changing the setup implies a great deal of financial cost in most cases.

Performance of a system is not something that is directly and easily obtainable. This is because systems are comprised of smaller components, each of which should be studied separately, as well as thier interconnection.

When a study of the performance of an operating system of a computer system is done, many factors should be taken into account and at a different level of detail.

When a performance study is being conducted, the information needed could either be obtained from the system itself by using the measurement techniques, or from a model of that system by using the modeling techniques.

Measurement is an accurate method to get information about a specific system. It gives credible results, but has the disadvantage of not being feasible sometimes. This occurs if the system is still under design, or it is

extremely expensive to initiate a measuring experiment. It is also not very effective, since results obtained may be too late to be of any use. Modeling on the other hand, should be used mainly when the system is not available. The unavailibility of the system might occur due to the expense, danger, or nonexistance of the system under consideration.

Some of the methods and techniques used for performance evaluation will now be briefly described.

2.2. Measurement techniques.

Some basic measurement techniques include, hand timing, audiovisual aids, and benchmarking which is performing some specific experiments on the system, and copmaring the results with experiments held on other systems.

Measurement techniques could be used to tune systems for optimal performance once adequate performance has been obtained. Measurement techniques might not be the best to study the performance of systems with unknown performance.

When studying the performance of a system by using measurement techniques, it must be decided what to measure and how to measure it. This is usually done by using tools or instruments. Experiments should be

carefully, and correctly designed to come up with meaningfull, and credible performance information.

Measurement Tools.

A measurement tool is expected usually to sense the occurance of a specific situation, perform some transformation on the measured data, and report some information reflecting the results of the measurement. Care should be taken when choosing a specific tool to be used for a specific experiment. Relevance of the tool to the measured data should be guaranteed and the appropriate tool should be chosen for use. Accuracy of the tool should also be checked to obtain data of the required detail.

There are different classes of tools used to measure performance of a computer system, namely hardware, software, and firmware tools. There are hybrid tools however, combining the properties of some of these classes.

Choosing the best tool to use is not a simple task and requires some experience. Different tools produce different granularity of results at different costs. The more involved and accurate the results are requested, the more it costs to incorporate the tool. For example, placing some debugging statements into the software might be very costly since this might overload the system as a whole. Therefore, chosing a testing tool should be done

after a detailed study of the consequences.

Experiment Design.

After choosing the appropriate tools for the study, the experiment should carefully be designed. The stages of experiment design go from problem identification and definition to the selection of the level and duration of experimentation. After tests are carried out, results should be interpreted. This last stage is an important and difficult part of any performance evaluation study.

Presentation of Results.

Results could be presented in different ways. They could be given as numbers, which would be difficult for the reader to comprehend. They could be represented as Gantt charts, or Kiviate graphs, which are used to graphically summarize the utilization charachteristics of the system's components. Other presentation methods also exist [5,7].

Result Analysis

Results should then be analysed by using several methods such as variance analysis, regression analysis, etc. The analysis should be able to indicate possible problems the system is facing.

If the performance of a computer system is to be

considered in the design and development stages, it will not be possible to carry out any experiments on the system. Modeling must then be used to estimate the performance of the proposed system.

Building models for systems under design does not imply that this method could not be used to model existing systems. On the contrary, modeling can also be used to test different setups of existing systems, such as an increase of main memory or disk space, and impose costly and expensive situations, that if imposed on the real system, might lead to unpredictable results.

When a system is designed with no regard to performance, and it shows unacceptable performance later on, it might be necessary to add some hardware to make up for the deficiency, or go through redesign, and redevelopment phases. In both cases, it costs much more than a design with special consideration for the expected performance.

2.3. Modeling Techniques.

Some modeling techniques include simple formulae building, numerical scoring methods, kernel programs, synthetic programs, analytic models, including statistical models, graphical models, algorithmic models, markovean queueing models, and simulation models.

In this chapter, an overview of analytic and simulation models will be given. [7]

Model Building.

The model of a system is a representation of that system consisting of certain amounts of organized information about it, and is built for the purpose of studying it [2].

The model is built by extracting the main factors determining system performance, determining performance measures in the model, and using these measures as estimates of actual system performance [1].

While building a model of any system, one should develop a certain evaluation function that would include several parameters each of which has a specific weight depending on the parameter's effect on the system performance. Development of such a function is not a trivial task.

The modeler should be able to judge the effect of different system parameters on the overall system performance, then according to the order of parameter weights, the modeler is going to incorporate them in the model. There may be many system components that are important in the system function, but do not affect the performance to a great degree, therefore these parameters will not have any significant weight, even if they were

incorporated in the model. It is always best to keep the model as precise and simple as possible, so that it is managable and reflects the functions of the actual system.

When building the model, the modeler should produce in his mind what is called a Conceptual Model [2]. The deeper the modeler knows about the system, the better the conceptual model. From this conceptual model, the modeler is expected to produce the actual model, by using the general modeling techniques, such as the simulation techniques, or any of the analytic techniques.

Analytic Models.

Analytic models of computer systems are sets of mathematical equations whose independant variables (inputs) produce a single set of dependant variables (outputs). The need to make a model analytically solvable often imposes the introduction of simplifying assumptions that influence both the degree of detail, and the configuration of the model. There are basically two major types of analytic models, deterministic and probabilistic.

Deterministic models are not very suitable for modeling computer systems, especially dynamic systems, because they impose the need of discretyzing the model, which is not plausable. They might be used however to model worst-case or best-case conditions.

Probabilistic models are commonly used. They can model systems possessing characteristics such as continuous or discrete, deterministic or probabilistic, static or dynamic, linear or non-linear, open or closed, stable or unstable.

The mathematical model of a computer system may be in discrete state space and continous time, usually probabilistic, dynamic, non-linear, might be open or close, stable or unstable.

Queueing models.

Queueing models are used widely due to their ability to represent complex highly variable phenomena in a relatively compact manner. Queueing models are constructed by considering each relevant resource in the system as a server recieving requests from entities flowing in the system, these entities are system dependant. When the server is busy, the request has to join a queue until its turn comes to be serviced. In an information processing system for example, a request might be a program, a program step, an interavtive command, or an I/O request, depending on the degree of detail incorporated in the model.

Queueing models could be constructed as complex as needed, and they are simple to expand, since they use queueing theory principles.

A computer system could be easily represented as a queueing model, since there is a high resemblence between resources and servers, and between requests and customers in the queueing model.

A queueing system is usually studied after a transient period of time which eliminates unstable situations before reaching what is called the steady state of the system which represents the actual system behaviour.

Simulation Models.

Analytic models could actually be considered as special cases of simulation models. Analytic models are deterministic producing the same results no matter how many times they are solved. Simulation models on the other hand are non-deterministic probably producing slightly different results for different runs of the model.

Another point of difference need also be noted. Analytic models have to be solved to produce results, whereas, simulation models are run to produce results.

Reproducing the system behaviour according to some convention establishing a correspondance between the model and the actual system charecterizes simulation models. Simulation models are the most popular approach used to model computer systems, because of their

generality and ability to immitate the system in detail. There are however some noted disadvantages of this technique, such as the expense of model construction, which includes programming errors, and unavailibility of input data.

Another disadvantage might be the computational expense of running the simulation program. Statistical analysis of the program behaviour could also be lengthy and costly.

2.4. Using simulation models.

Studying performance using the simulation technique, is done if the system is unavailable, or if incorporating a measurement experiment is too cumbersume and not reliable. Simulation models are basically built to test a design hypothesis, or demonstrate performance and sensitivity of the proposed system.

The other major use of the simulation models is to evaluate performance of a system that exists, but is not available for experimentation. An example is the design of a computer system for weapon control. It is not feasible to build the system, incorporate it in the weapon, then test it live. It might be dangerous and costly. Thus a simulation model of the control system could be built and tested as a model before the actual system is tested on live ammunition.

Formulation and Construction of Simulation Models.

There are some basic issues that affect the formulation and construction of simulation models, some of these issues will be discussed briefly.

Degree of detail is one issue that should be decided from the begining of the model building stages. This tells the modeler to what level he should go in incorporating the factors of the real system in the model. There is always the question of how far to go on incorporating the details of the model to resemble and simulate the actual system. More the model incorporates deatails from the actual system, more it resembles the actual system, but on the other hand, more complex and unmanagable the model becomes.

The art of model building appears at this stage. What to choose as factors from the actual system that would build up a good model, and still keep it simple and managable, is the main issue. There might be factors that seem to be affecting system operation, but do not actually reflect as a high percentage in the overall system performance.

Flexibility is an important issue affecting the formulation of the model, since it is always wise to assume that the system being modeled is subject to change, so that several setups and design changes could be tested.

The simulation language used in building the model has a direct affect on the way the model is to be designed. There are languages that use special design methods, such as the Information Processing Graphs (IPG) facility of the Performance Analyst Workbech System (PAWS) simulation language, where the system could be represented graphically, then coded using PAWS to produce the simulation program. This would incline the modeler to adopt a specific view of the system. The PAWS simulation language is discussed in detail in chapter 3.

There are a veriety of simulation languages ranging from FORTRAN programs, which allow flexibility, and incorporation of minute details in the model, to the very high level simulation languages such as PAWS, which use high level routines, that would help the modeler concentrate on building the model rather than worrying how to implement specific modeling details.

The Process of Simulation.

The simulation process could be summarized as follows. First the problem is defined, and the need for building a simulation model is validated. Selecting the solution method follows, then the development of the model can begin. When the design is over, an appropriate simulation language is chosen and the implementation can start. When the model is built it should be validated and verified. After which several simulation runs

are done to extract information needed to perform the statistical study of the system. Documentation of the results and findings of the modeling process is needed, to pass information about the performance of the system to higher management.

One of the major drawbacks of building simulation models, is the high cost of initial model building, so one time use models are rarely justified.

It is highly recommended to build the model in the design stages of the system, and to build it in a manner that could incorporate the developments imposed on the system in future. The model should be built to be used continuosly, so that its cost can be justified.

When the cost of model building is incorporated in the design costs, it would insure, to a great deal, the acceptable performance of the initial setup of the designed system. The model could then be used to fine tune the system, and reach optimum performance at minimal cost.

3. PERFORMANCE ANALYST'S WORKBENCH SYSTEM (PAWS)

3.1. Introduction

Building a simulation model requires the use of a simulation language. There are however several criteria used to choose a certain simulation language for building a specific model. Features that are expected of simulation languages include the ability to generate random numbers, ability to handle queues, ability to adapt to the nature of the modeled system, vulnarability to change, and statistical analysis.

The Performance Analyst's Workbench System (PAWS) was originally designed by the Information Reaserach Associates of Austin Texas in 1983 to model computer systems. PAWS design goal is clear from some of the basic constructs in PAWS that will be described in this chapter. Availibility of these constructs is the basic reason for choosing PAWS as the simulation language to be used for implementing the SAMIS Online system model.

PAWS is a simulation language in which a pictorial representation of a model is directly translated and evaluated by a computer. PAWS has two important features, basing model building on pictorial representation, and supporting a number of high level primitives such as bestfit memory management, FCFS

queueing discipline, or exponential probability distribution. These characteristics allow the modeler to concentrate more on the high level of modeling rather than the low level of writing code for such standard primitives.

Memory for instance, seen as contiguous storage locations, incorporated in the basic design of PAWS, solves a lot of modeling problems. Building a model of memory, using for example the best fit memory management method, in a traditional simulation language such as GPSS is not a trivial task. Doing it in PAWS might take only a couple of lines.

The ability to design special nodes, where the designer could implement an algorithm in a different programming language in the model whenever neccessary is a major advantage of PAWS. This newly designed node is treated by PAWS as any other node, where control can go in and out of that node, and branches could be made to and from it. This feature makes PAWS very powerfull and versatile.

Some of the most commonly encountered queueing and memory management techniques have been incorporated in the basic PAWS package. Nevertheless other special techniques could be implemented through the use of the special user node in a FORTRAN 77 interface.

These features, and many more made the choice of PAWS

look not only natural, but necessary for our purposes.

3.2. Development of a Model in PAWS.

A modeling project using PAWS would proceed by first obtaining possible parameters affecting system performance from the system design, be it proposed or existing. Constructing and evaluating a PAWS model to obtain system performance estimates follows. This construction is aided by the use of an Information Processing Graph (IPG) which is a modeling methodology using pictorial representation of information processing.

The model design is then directly translated to a simulation program using the PAWS simulation language. Several runs of the PAWS model are then made, each of which corresponds to a system design variation. Statistics obtained from these runs can then be used to choose or recommend the most desirable set up, and could be used as a basis for future design or configuration refinement. Such a modeling cycle is depicted in Fig. 3.1.

Information Processing Graphs.

An information processing graph (IPG) is a pictorial construct used for modeling information processing systems. It shows the interconnection of a set of nodes, that process the information passed to them, and produce certain actions. Such nodes might be CPUs, disk units, etc. Information being processed might represent transactions, tasks, etc.

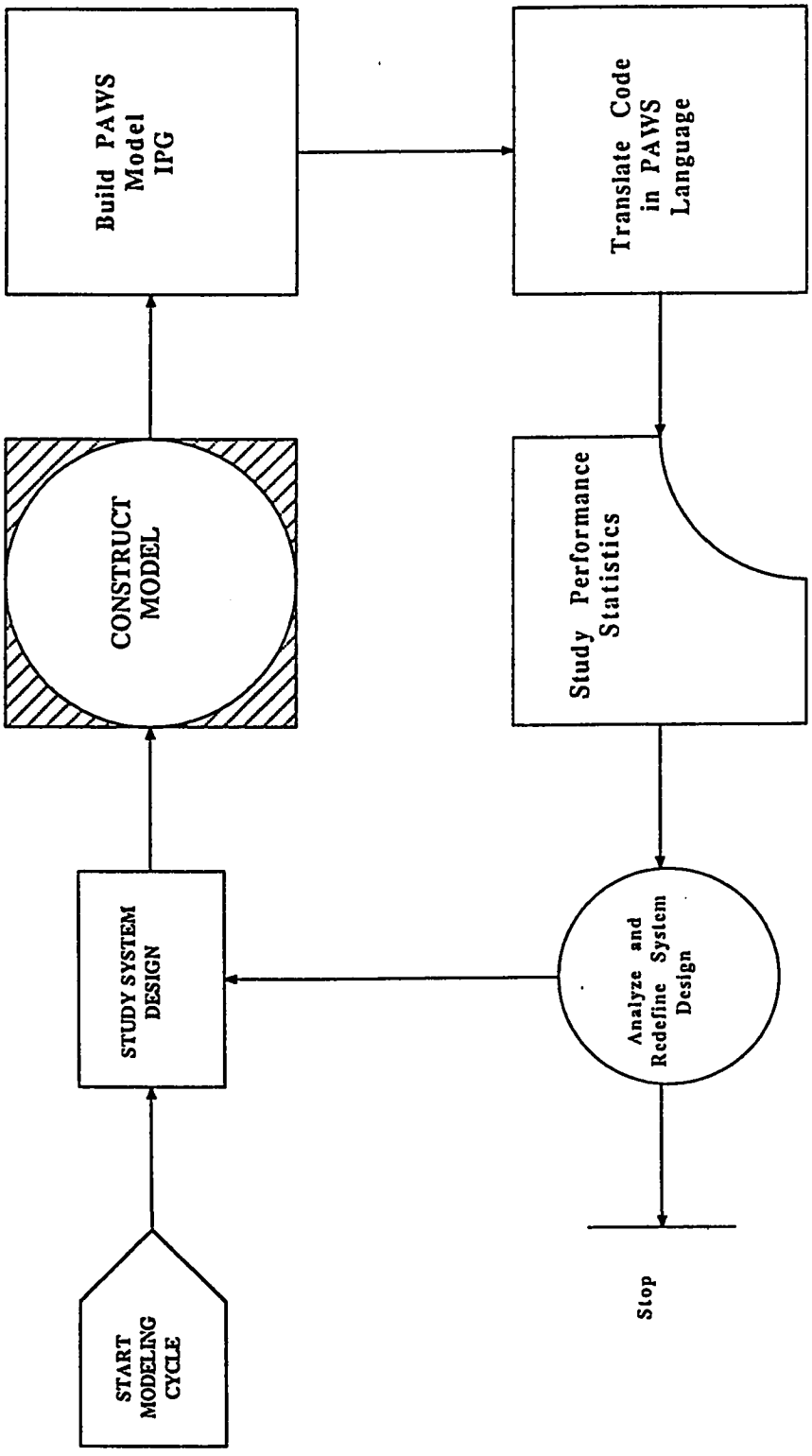


Figure 3.1 Modeling Cycle Using PAWS

The IPG is used to describe the information flow as well as to generaly specify the processing to be done at specific nodes. Nodes in an IPG represent a resource, a routing decision, or a computational step. An edge connecting the two nodes represents a form of information flow from one node to another. Selective flow of transactions from one node to the other is done by labeling the edges with transaction category, transaction phase, and branching probability.

Transactions are the basic entities being processed and routed in an IPG. They basically represent the data on which the nodes of an IPG operate on. Transactions might actually be a job, a person, or any other basic entity in information processing systems.

A simple IPG is shown in Fig. 3.2 where some transactions enter the IPG from the node 'enter', and they stay in a closed loop between the CPU and the two disk units. Transactions entering this system stay in the CPU-Disk cycle for ever, and transactions going from the CPU have 50 percent probability to go to disk 1, and a 50 percent probability to go to disk 2. All transactions leaving both disks are routed back to the CPU again.

A computer evaluation of such a model giving performance statistics, requires the translation from this pictorial representation to a computer understandable

code. This is done using a direct translation mechanism of the PAWS simulation language.

PAWS language provides a convenient notation for describing information processing graphs. A PAWS program completely describes the IPG structure as well as the individual characteristics of each node. The PAWS simulator then reads and evaluates the model written in the PAWS language to obtain performance statistics for the model. Fig. 3.3 shows the files used and produced by the PAWS simulator. The statistics file produced (.STA) along with other output files are studied before the analysis and redefinition phase is conducted to make appropriate recommendations.

3.3. The PAWS Language.

A PAWS program is constructed of the following parts in sequential order [10].

1. OPTIONS part (optional), specifies an optional format, the only so far.

2. DECLARE part, declaring all variables, nodes, categories, tokens and memories.

3. TOPOLOGY part, describing the structure of the model, which describes how information and control flow in the system depending on certien conditions. This section shows the interconnection of nodes in the PAWS model.

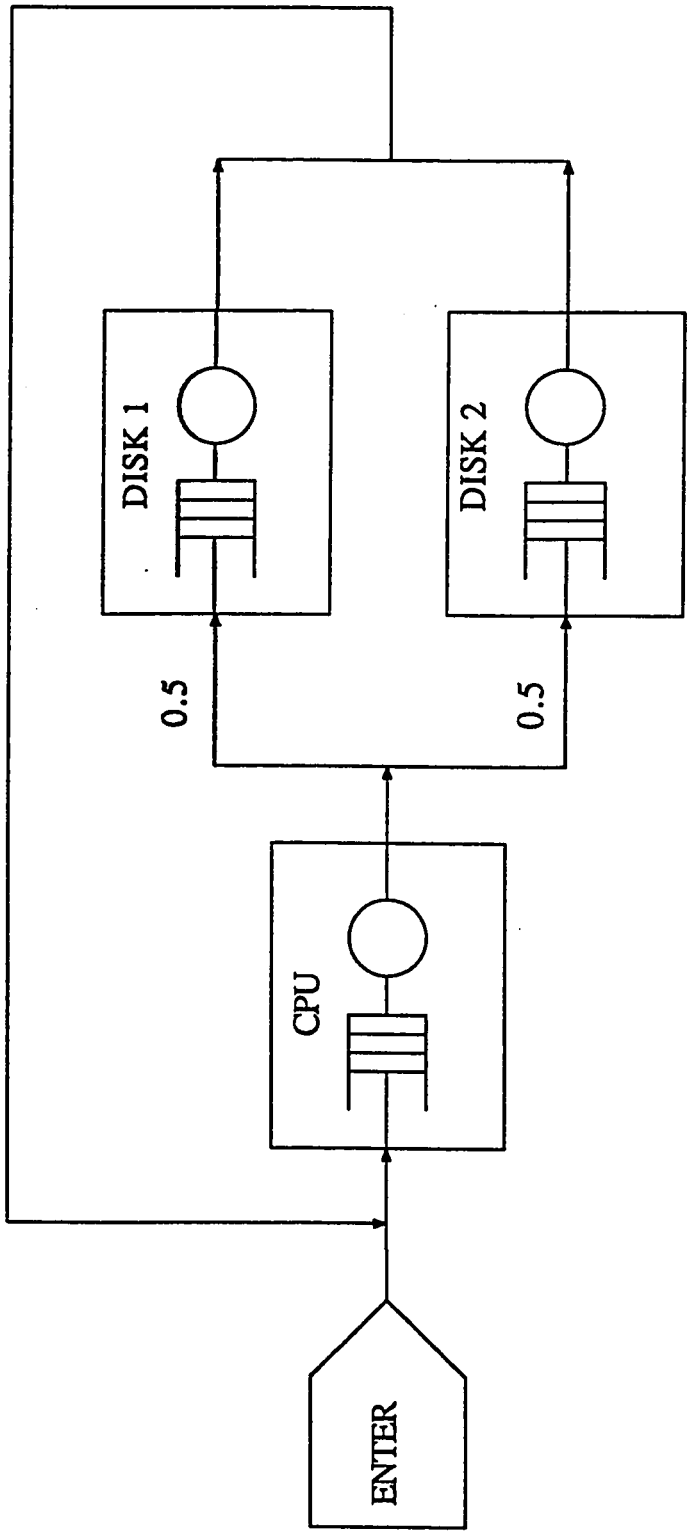


Figure 3.2 : Simple IPG

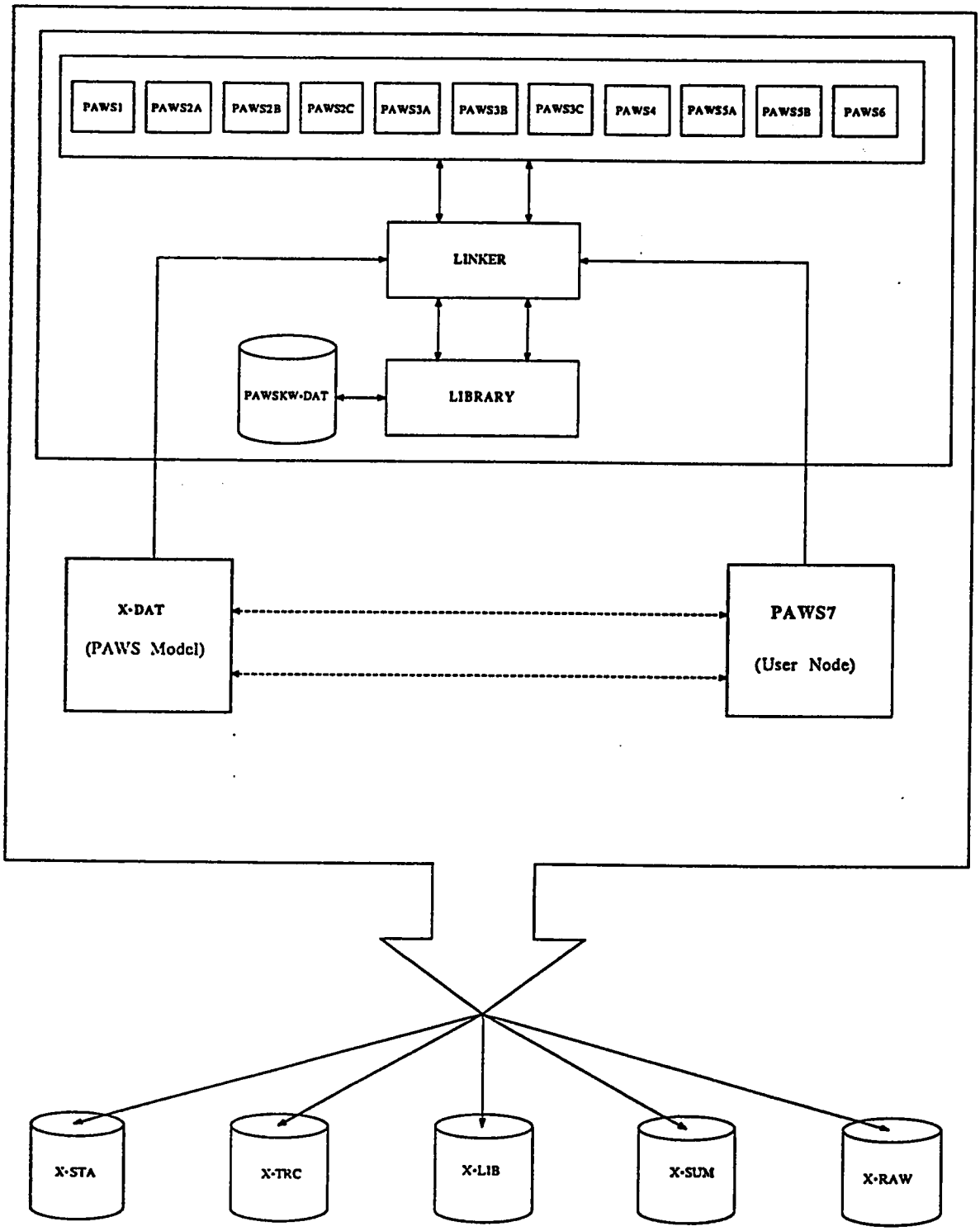


Figure 3.3 : Files Used and Produced

by the PAWS Simulator

4. DEFINE part which describes the function of each node specified in the TOPOLOGY part(whether these nodes are service nodes, compute nodes or user nodes, see node definition later on).

5. INITIAL part (optional), which describes the initial state the system is to be in when it is first operated, and whenever a reset is done.

6. STATISTICS part (optional) specifies what statistics are to be printed, and to what detail.

7. RUN part (optional) specifies detailed description of simulation control (timming, reporting statistics, simulation state, random number generation,...)

Each section of a PAWS program starts with a keyword and ends with a semicolon. PAWS programs are free formatted, with the exceptions that any characters beyond column 70 are ignored, and that all characters following a (!) or (%) are also considered as comments and ignored. The entire program is terminated by the "END;" statement.

Time units are user defined, and once the time unit is set it will always be used by the PAWS simulator to produce performance statistics. The modeler should be consistant in the use of the time units.

All the nodes in the model should be declared in the

DECLARE section, then in the define section the type of each node, whether it is a resource management node, routing node, arithmetic node, interrupt node, or a user node should be specified. A general description of all types of nodes in a PAWS model follows [10].

Resource Management nodes.

There are actually two types of resources that could be modeled by PAWS. These are resources that actually do the work, and resources that just have to be obtained by the transaction to be able to proceed and get processed. These types of resources are respectively called ACTIVE, and PASSIVE resources.

Active Resources

An active resource, such as a CPU, should be held by the transaction and should do some work for the transaction for a specified amount of time, depending on the scheduling and management techniques used. The time the resource is held does not in general depend on any other outside factor.

Passive Resources.

A passive resource, such as a memory segment, will be held by a certain transaction for an amount of time that is not specified, and does not follow a certain scheduling or management technique. It does however depend

on other factors, such as how many CPU and I/O cycles are going to be done by the transaction before the memory is released, and made accessible to other transactions.

Active Nodes.

An active resource is described in an IPG as an active node, which is represented in PAWS as a SERVICE node. An incoming transaction requiring 'service' from the resource node has to wait in a queue if the resource is busy. The type of waiting need be defined by the modeler according to the specifications of the modeled system. The transaction then enters the 'server', and receives service for a specified amount of time, again predetermined by the modeler (probably non-deterministicly), quits the service node and proceeds according to the paths outgoing from this node, still keeping all passive resources that should be held for processing, i.e. pages in memory that is required to be held by any transaction being processed by a CPU. When the active node is entered, the active resource is possessed, then when the node is left, the active resource is released.

Passive Nodes.

A passive resource is first acquired and later is released when a certain condition happens, which might take some unknown amount of time, and after going through a sequence of steps, among several nodes in the system.

There are two types of passive resources and the acquisition and release of each type are performed at two different nodes.

Tokens and Memories.

Passive resources are either single resources, i.e. a channel, I/O buffer, or groups of contiguous resources of similar nature i.e. main memory space. Single passive resources are represented as TOKENS, whereas groups of contiguous passive resources are represented as MEMORIES.

ALLOCATE,RELEASE, and GETMEM,RELMEM nodes.

The nodes that indicate the acquisition and release of TOKENS are ALLOCATE and RELEASE, whereas the nodes that indicate the acquisition and release of MEMORIES are GETMEM and RELMEM. A TOKEN, when released, usually but not necesserily is sent back to the node that allocated it.

When the resource, be it active or passive, is not available, busy, or used, the incoming transaction has to know what is to be done. Usual criteria include, aborting, looking for another resource, getting the resource regardless whoever being serviced by interrupting the current transaction, and waiting until whoever is being serviced quits.

PAWS was made to accomodate some of the most commonly

used criteria. The queueing disciplines that are provided by PAWS include First Come First Serve, Round Robin Fixed Quantum, Processor Sharing, Polling, None Pre-emptive Prioroty, Pre-emptive Priority, Last Come First Serve with Pre-emptive Resume, and Delay.

Power of a SERVICE node.

The rate of service provided by any resource (server) can dynamically be modified by other transactions in the model. This concept can be used to model degraded or upgraded performance of a server. The power of a server is normaly set to one and rate of service can be increased by setting the power to a number greater than one, or decreased by setting it to less than one. A setting of power to zero represents the "off" state of a server.

Category and Phase.

Transactions flowing in a system do not always act the same way, for example a transaction representing an operator think time waits in a service node much longer than a transaction waiting for a disk I/O to be complete, same as a student in art school graduating way before a student in medicine school, although they are both students, but they behave diffrently because they are of different categories.

The concept of categories of transactions was incorporated in PAWS. Each transaction has a

category. The category of a transaction does not change during the lifetime of the transaction. Those aspects of a transaction that are variable in nature during the lifetime of a transaction are represented by the phase of transaction.

Routing Nodes.

Routing nodes in PAWS can be classified in six different types [10].

1. SOURCE nodes: nodes that produce or create initial transactions.

2. SINK nodes: transactions die when they reach a sink node.

3. FORK nodes: where a parent transaction gives birth to some children transactions, and waits until they get together again in a JOIN node, to ensure parallelism and synchronization.

4. JOIN nodes: where children join up to recreate their parent, and die.

5. SPLIT nodes: where a parent gives birth to children, and continues on,which produces the effect of parallelism, but not sunchronization, used for message broadcast for example.

6. BRANCH nodes: a convenient way to economize PAWS code,

and to help in statistics collection.

Arithmetic and Change nodes.

A third type of node in an IPG is an ARITHMETIC node, where some simple algorithms could be implemented. A COMPUTE node is one of the two types of possible arithmetic nodes, the other being the CHANGE node.

In a compute node, aritmetic and logical operations can be CARRIED OUT. In a CHANGE node, the phase of a transaction can be changed.

Interrupt nodes.

Sometimes while the system is running, a special case occurs, and some specific types of transactions are alerted by some other types of transactions, then these alerted transactions, are routed to a certien path according to the nature of the case. To implement this concept in PAWS, an INTERRUPT node was devised. A transaction entering this node interrupts some other specified set of transactions, aborts their wait in the queue, changes their phase, and sends them to other pre-specified nodes [10].

USER nodes.

Most simulation languages are not expandible in the sense that they donot allow a modeler to incorporate programs developed in other programming languages into the

model. This feature is necessary since the modeler may find the facilities provided by the simulation language insufficent.

PAWS provides such a feature through a special node called the USER node. This node transfers control to a FORTRAN subroutine, whose name is USS, that implements a certain algorithm. In a USER node, properties of a transaction, such as its phase, can easily be modified. Control is then returned to PAWS, which proceeds according to the new set of states reached after the subroutine was executed.

Statistical Analysis.

To obtain statistics about the performance of the model, the user requests, in the STATISTICS section of the PAWS program, the production of several statistics on the queue length or queueing time for a specific node in the model. These nodes should be either ALLOCATE, GETMEM, SERVICE nodes, or any array of such nodes. Response time statistics between two nodes can also be requested. Statistics are given in the form of numbers as well as histograms. It is possible to specify the histogram intervals.

The PAWS simulator produces several output files including (See Fig. 3.2) :

1. A compilation listing file, listing the PAWS program

with any possible compilation errors.

2. A detailed statistics file, containing reports about the throughput, queue lengths, queue times. response times, and utilization statistics.

3. A summary statistics file, summarizing the detailed statistics file.

4. An event trace file, listing brief descriptions of events simulated during simulation time if the trace flag was enabled.

5. Raw statistics file, listing any values specified by some nodes, like the COMPUTE node.

6. An interactive output file, containing periodic notice messages generated.

The study of the detailed statistics file gives the modeler the information needed to produce a performance measure of different elements of the model. Information in the detailed statistics file is grouped and reported according to transaction category. Overall statistical report of the behaviour of all transactions, is also provided.

Statistics collected by the detailed statistics file include the following [10] :

- Throughput count : count of transactions departing a

node in the model.

- Throughput rate : throughput count divided by simulated batch duration.
- Input count : count of transactions arriving at a node in the model.
- Input rate : input count divided by simulated batch duration.
- Queue length : the number of transactions (grouped according to category) waiting or receiving service at a node in the model at the time of statistics reporting.
- Mean queue length : time weighted average queue length at a node for a transaction category during the time of statistics reporting.
- Queueing time : on a specific node is the simulation time the transaction leaves the node, minus the simulation time when the transaction had arrived at the node.
- Mean queueing time : on a certain node is the average queueing time of all transactions in a certain category during the batch the statistics were reported.
- Observed service time : amount of transaction's service request given to the transaction, in a SERVICE node.
- Mean observed service time : average service time for all transactions of a certain category during the statistics reporting time.

- Utilization percentage : throughput rate multiplied by mean observed service time multiplied by a 100.
- Response time : from node A to node B, is the simulation time the transaction leaves node B minus the simulation time when the transaction had entered node A.
- Mean response time : between nodes A and B is the average of all response times for all transactions of this category passing from node A to node B.
- Mean number of (T) tokens held : is the number of type (T) tokens held by all transactions of this category during the batch the statistics were reported.
- Utilization percentage of token (T) : is the time weighted average percentage of existing (T) tokens held by transactions of this category.
- Utilization percentage of memory (M) : is the time weighted average of memory (M) elements held by the transactions of this category during the batch the statistics were reported.

Statistics are presented in groups in the detailed statistics file, in the following sequential order.

1. Basic statistics, including the number of events simulated during the statistics report period, batch number, batch duration, batch start and dump time.

2. Throughput statistics, includes throughput rate and count for each category, and for each node. Input

rate and count is also provided here for each category, and for all BRANCH, ALLOCATE, GETMEM, and SERVICE nodes.

3. Queue statistics, include a summary on each group and each ALLOCATE, GETMEM, and SERVICE nodes. Mean, second moment, standard deviation, variance of the queue length, and queue time is also printed. If requested in the STATISTICS section, queueing length and queue time histograms are provided.

4. No activity statistics, include listings of all nodes followed by names of categories of transactions that did not leave each node during the batch period the statistics were reported.

5. Server statistics include the mean observed service time, and the utilization for each category and each SERVICE node in the model.

6. Token statistics include token utilization, and mean number of tokens held for each category and each token type.

7. Memory statistics include memory utilization for each category and each memory in the model.

8. Response time statistics are printed according to the user specification in the STATISTICS section. A representative histogram and summery data will be

printed.

9. All variable values including scalar and global variables when the dump occured will be printed.

Simulation Control

The initiation and termination of simulation batches are done in the RUN section of a PAWS program, along with the collection and reporting control of statistics and treatment of simulation states between simulation batches and during simulation PAUSEs.

Several operations could be performed within the RUN section, including most of the operations that could be done in the COMPUTE node. Statistics could be dumped on request and the model state could be RESET or CLEARed at any time. CLEAR is useful when transient statistics are not needed, and should be discarded.

Setting the simulation duration is the most important feature of the RUN section. Notice interval could be specified where the model gives notices about the simulated time in advencements of notice intervals.

3.4 Example PAWS Model

In this section, an example of model construction using PAWS will be illustrated through the discussion of a simple system that is going to be modeled using PAWS.

Model constuction with the help of PAWS goes through several stages. First the study of the system should be done, then the construction of the model begins by developing a conceptual model, and constructing the information processing graph, then that is translated to PAWS code, and the statistics are studied. In this example, we will stop at the stage of obtaining the statistics becuase statistics interpretation needs more insight and detailed explanation.

Consider a batch system where jobs arrive on the average once every 0.2 seconds. Jobs queue up in a first come first serve queue waiting for memory to be allocated to them, according to a first fit memory management techniqe. The memory consists 500 1K-word blocks, and the memory requests per job is uniformly distributed between ten to fifty consecutive blocks. When obtaining the memory the job will proceed to the CPU where it will be queued up for service according to a round-robin fixed-quantum scheduling discipline, with quantum of ten milliseconds. CPU burst times are hyperexponentially distributed with an average of ten milliseconds and a standard deviation of 14.1 milliseconds. Upon completion of the CPU burst the job will be queued up for I/O service on the disk unit, the disk serves first come first serve with a service time uniformly distributed between ten to twenty milliseconds. Upon completion of the disk I/O, the job either returns to the CPU for another

CPU cycle, or it releases the memory, and exits the system. The job makes an avearge of ten CPU-to-Disk cycles before leaving the system. [10]

The information processing graph (IPG) for this example is shown on Fig. 3.4 and the PAWS language implementation is in Fig. 3.5. PAWS output files are shown in Fig. 3.6 to Fig. 3.9.

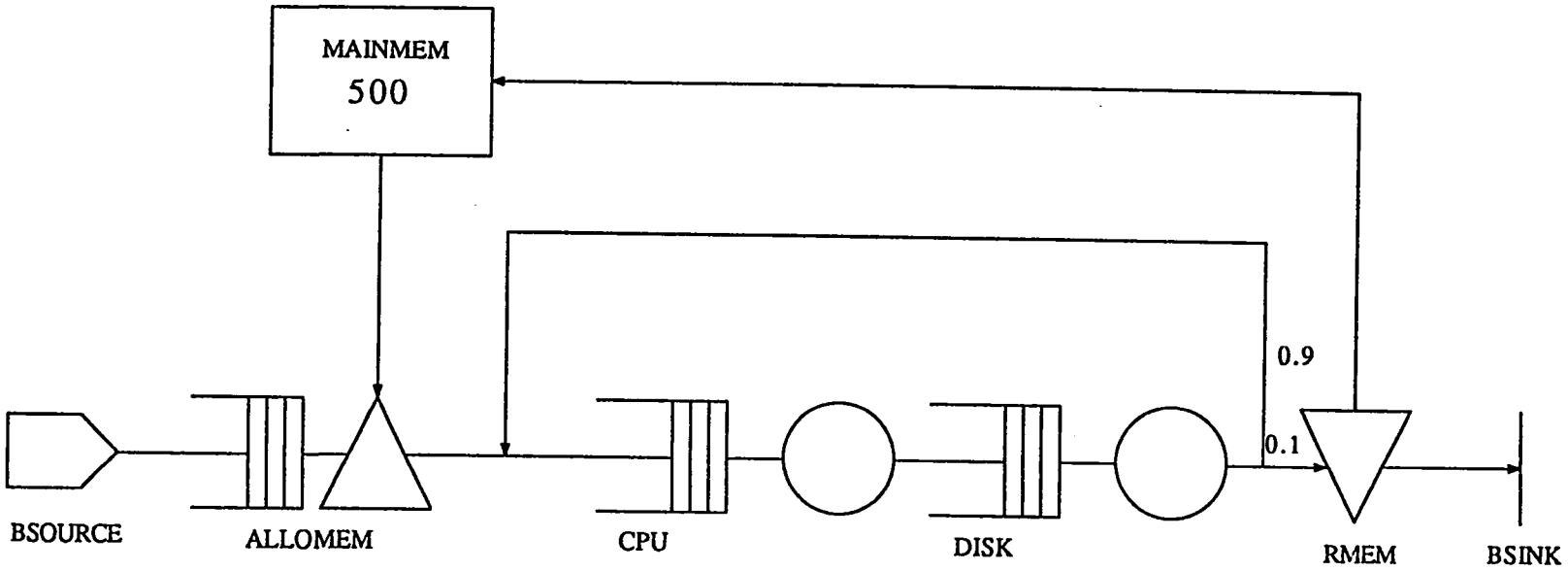


Figure 3.4 : Example IPG

```
PAGE  1      I R A   P A W S   -   V   2 . 0   *   C O M P I L A T I O N


000001  DECLARE
000002    NODES      BSOURCE, ALLOMEM, CPU, DISK, RMEM, BSINK;
000003    CATEGORIES BATCH;
000004    MEMORIES   MAINMEM;
000005
000006  TOPOLOGY
000007    BSOURCE     ALLOMEM     (BATCH,ALL)      1.0;
000008    ALLOMEM     CPU         (BATCH,ALL)      1.0;
000009    CPU         DISK        (BATCH,ALL)      1.0;
000010    DISK        CPU         (BATCH,ALL)      0.9;
000011    DISK        RMEM        (BATCH,ALL)      0.1;
000012    RMEM        BSINK       (BATCH,ALL)      1.0;
000013
000014  DEFINE
000015    BSOURCE
000016      TYPE SOURCE
000017      REQUEST (BATCH,1) EXPO(200.0);
000018    BSINK
000019      TYPE SINK;
000020    ALLOMEM
000021      TYPE GETMEM
000022      QUANTITY 500 MAINMEM FIRSTFIT
000023      QD FCFS
000024      REQUEST (BATCH,ALL) UNIFORM(10,50) MAINMEM LI[3];
000025    CPU
000026      TYPE SERVICE
000027      QUANTITY 1
000028      QD RRFQ 10.0
000029      REQUEST (BATCH,ALL) HYPER(10.0,14.1);
000030    DISK
000031      TYPE SERVICE
000032      QUANTITY 1
000033      QD FCFS
000034      REQUEST (BATCH,ALL) UNIFORM(10.0,20.0);
000035    RMEM
000036      TYPE RELMEM
000037      REQUEST (BATCH,ALL) MAINMEM LI[3];
000038  STATISTICS REPORT
000039    QT CPU  GRAN 7.0;
000040  RUN
000041    GO 100000.0 2000.0  CLEAR
000042    GO 400000.0 2000.0  DUMP
000043    EXIT;
000044  END;
```

Figure 3.5 : Example PAWS Program

PRODUCED BY:
PAWS 2.0 - PERFORMANCE ANALYSTS WORKBENCH SYSTEM,
INFORMATION RESEARCH ASSOCIATES, AUSTIN, TEXAS.

PAGE 2 I R A P A W S - V 2.0 * S U M M A R Y S T A T I S T I C S P A R T 1

*** BATCH NUMBER: 2 BATCH DURATION = 300000.000 (FROM 100000.000 TO 400000.000) ***

		THROUGHPUT		QUEUE LENGTH				QUEUEING TIME			SERVICE	
		RATE	COUNT	MIN	MEAN	MAX	END	MIN	MEAN	MAX	MEAN	UTIL
NODE: BSOURCE	(1)											
CAT: BATCH		0.005	1442.									
CAT: ALL		0.005	1442.									
NODE: ALLOMEM	(1)											
CAT: BATCH		0.005	1442.	0	0.000	1	0	0.000	0.000	0.000		
CAT: ALL		0.005	1442.	0	0.000	1	0	0.000	0.000	0.000		
NODE: CPU	(1)											
CAT: BATCH		0.048	14461.	0	0.830	14	3	0.000	17.201	517.031	10.130	48.828
CAT: ALL		0.048	14461.	0	0.830	14	3	0.000	17.201	517.031	10.130	48.828
NODE: DISK	(1)											
CAT: BATCH		0.048	14462.	0	2.193	14	0	10.000	45.498	210.969	14.980	72.213
CAT: ALL		0.048	14462.	0	2.193	14	0	10.000	45.498	210.969	14.980	72.213
NODE: RMEM	(1)											
CAT: BATCH		0.005	1440.									
CAT: ALL		0.005	1440.									

Figure 3.6 : Example Statistics (1)

PAGE 1 I R A P A W S - V 2 . 0 * S T A T I S T I C S

PRODUCED BY:
PAWS 2.0 - PERFORMANCE ANALYSTS WORKBENCH SYSTEM,
INFORMATION RESEARCH ASSOCIATES, AUSTIN, TEXAS.

PAGE 2 I R A P A W S - V 2 . 0 * B A S I C

NUMBER OF EVENTS: 53308.000
BATCH NUMBER : 2
BATCH DURATION : 300000.000
BATCH STARTED AT: 100000.000
CURRENT TIME : 400000.000

PAGE 3 I R A P A W S - V 2 . 0 * T H R O U G H P U T

	THROUGHPUT RATE	THROUGHPUT COUNT		INPUT RATE	INPUT COUNT
NODE: BSOURCE (1)			*		
CAT: BATCH	0.005	1442.	*		
CAT: ALL	0.005	1442.	*		
NODE: ALLOMEM (1)			*		
CAT: BATCH	0.005	1442.	*	0.005	1442.
CAT: ALL	0.005	1442.	*	0.005	1442.
NODE: CPU (1)			*		
CAT: BATCH	0.048	14461.	*	0.048	14464.
CAT: ALL	0.048	14461.	*	0.048	14464.
NODE: DISK (1)			*		
CAT: BATCH	0.048	14462.	*	0.048	14461.
CAT: ALL	0.048	14462.	*	0.048	14461.
NODE: RMEM (1)			*		
CAT: BATCH	0.005	1440.	*		
CAT: ALL	0.005	1440.	*		
NODE: BSINK (1)			*		
CAT: BATCH	0.000	0.	*		
CAT: ALL	0.000	0.	*		

Figure 3.7 : Example Statistics (2)

```
QUEUE AT NODE: ALLOMEM    ( 1)
  CATEGORY: BATCH                                                    *** ALLOMEM    ( 1) ***
            QUEUE-LENGTH
              SUMMARY
                    MEAN:       0.000 2ND MOMENT:       0.000
                    VAR:        0.000 STNDRD DEV:       0.000

            QUEUEING-TIME
              SUMMARY
                    MEAN:       0.000 2ND MOMENT:       0.000
                    VAR:        0.000 STNDRD DEV:       0.000

QUEUE AT NODE: CPU        ( 1)
  CATEGORY: BATCH                                                    *** CPU        ( 1) ***
            QUEUE-LENGTH
              SUMMARY
                    MEAN:       0.830 2ND MOMENT:       2.082
                    VAR:        1.393 STNDRD DEV:       1.180

            QUEUEING-TIME
              INTERVAL                    NUMBER IN     % IN
                                          INTERVAL      INTERVAL      HISTOGRAM
                                                                0  10  20  30  40  50  60  70  80  90  100
        0.000 (= X (      7.000           6257.000      43.27   I**********************(                  I
        7.000 (= X (     14.000           3460.000      23.93   I***********(                              I
       14.000 (= X (     21.000           1618.000      11.17   I*****(                                    I
       21.000 (= X (     28.000            788.000       5.45   I**(                                       I
       28.000 (= X ( *INFINITY*           2338.000      16.17   I********(                                 I
                                         ..........                                                       I
                                TOTAL:   14461.000

              SUMMARY
                    MEAN:      17.201 2ND MOMENT:    1106.388
                    VAR:      810.510 STNDRD DEV:      28.469

QUEUE AT NODE: DISK       ( 1)
  CATEGORY: BATCH                                                    *** DISK       ( 1) ***
            QUEUE-LENGTH
              SUMMARY
                    MEAN:       2.193 2ND MOMENT:      10.426
                    VAR:        5.615 STNDRD DEV:       2.370

            QUEUEING-TIME
              SUMMARY
                    MEAN:      45.498 2ND MOMENT:    3243.764
                    VAR:     1173.680 STNDRD DEV:      34.259
```


Figure 3.8 : Example Statistics (3)

```
PAGE   5       I R A   P A W S   -   V   2 . 0   *   N O   A C T I V I T Y   S T A T I S T I C S


NODE: BSINK      (  1)
  CAT: BATCH
```

```
PAGE   6       I R A   P A W S   -   V   2 . 0   *   S E R V E R   S T A T I S T I C S


QUEUE: CPU        (  1) NUMBER OF SERVERS:          1
  CATEGORY      MEAN SERVICE TIME         UTILIZATION
  BATCH              10.130                    48.83
  ALL                10.130                    48.83
QUEUE: DISK       (  1) NUMBER OF SERVERS:          1
  CATEGORY      MEAN SERVICE TIME         UTILIZATION
  BATCH              14.980                    72.21
  ALL                14.980                    72.21
```

```
PAGE   7       I R A   P A W S   -   V   2 . 0   *   M E M O R Y   S T A T I S T I C S


MEMORY: MAINMEM
  CATEGORY       UTILIZATION
  BATCH                18.03
  ALL                  18.03
```

Figure 3.9 : Example Statistics (4)

4. The Saudi Arabian Ministry of Interior System (SAMIS)

4.1. INTRODUCTION

The Saudi Arabian Ministry of Interior System (SAMIS) is a computer based information system operated by the National Information Center (NIC), Ministy of Interior (MOI), Riyadh, to process information about the citizens and expatriates in the Kingdom of Saudi Arabia.

The SAMIS computer system contains two major components namely, the Online System, and the Administrative System. The Online System provides online information on passports, drivers licences, motor vehicles, civil registration, alien registration, and keeps record of miscreant, criminal records, boarder control and pilgrim control. The Administrative System is basically concerned with stock control, payroll, and such batch processed operations.

This research will be focused on the study of the Online System, due to the requirement of fast response time related to operations conducted by this system.

The Online and Administrative Systems run on different UNIVAC computer systems. A third UNIVAC computer system exists as a backup in case any of the two systems fails. The third system is also used for development and training purposes.

The SAMIS Online System runs on a UNIVAC 1100/82 computer system, and supports more than a thousand terminals connected to it in about two hundred and fifty sites located in different geographical areas within the Kingdom of Saudi Arabia.

The SAMIS Online System runs on an operating system called "Computer Science Teleprocessing System (CSTS-II)", developed by the Computer Science Corporation (CSC) Los Angeles, California. MANAGE is the data base management system used by the Online System to handle data bases under CSTS II.

The SAMIS Online System includes exactly eleven subsystems, each of which handles a spscific information processing requirement of the Saudi Arabian Ministry of Interior. These are Miscreant, Passport Issuance, Criminal Records, Alien Control, Boarder Control, Drivers License, Motor Vehicle, Pilgrim Control, Civil Registration, Micrographics, and Message Communication subsystems.

4.2 SAMIS Online System : Hardware Components

The SAMIS Online System runs on a dual CPU UNIVAC 1100/82 computer system. Over a thousand terminals are supported by use of various communication controllers and devices. The computer system has 1.5 Mega Words (one word =

36 bits) of main memory availabe, two disk subsystems, one drum subsystem, various tape devices, printers and other support equipment. All hardware components of the SAMIS Online System are shown in Fig. 4.1.

Main Memory

The size of the user area in main memory is 2282 pages, distributed as 1809 pages of Instruction Bank (I-Bank) and 473 pages of Data Bank (D-Bank). Memory is allocated to tasks by using bestfit. A page in the SAMIS system is 512 words of memory, and a word is 36 bits. See Fig. 4.2 for the map of main memory.

Disk subsystems

There are two disk subsystems, two storage control units per subsystem, and four strings of drives per subsystem. Each string has two channel adapter units connected to different storage control units to enable simultaneous access to two different drives in the same string. This means there is a total of 64 drives, consisting of 8 strings, therefore upto four concurrent disk i/o operations.

The minimum drive seek time is 10 msec, the max drive seek time is 55 msec, and the average latency is 8.3 msec. Drive capacity is 78,090 pages. Disk storage control unit to IOU channel-interface connections that are imbedded in the model are shown in table 4.1.

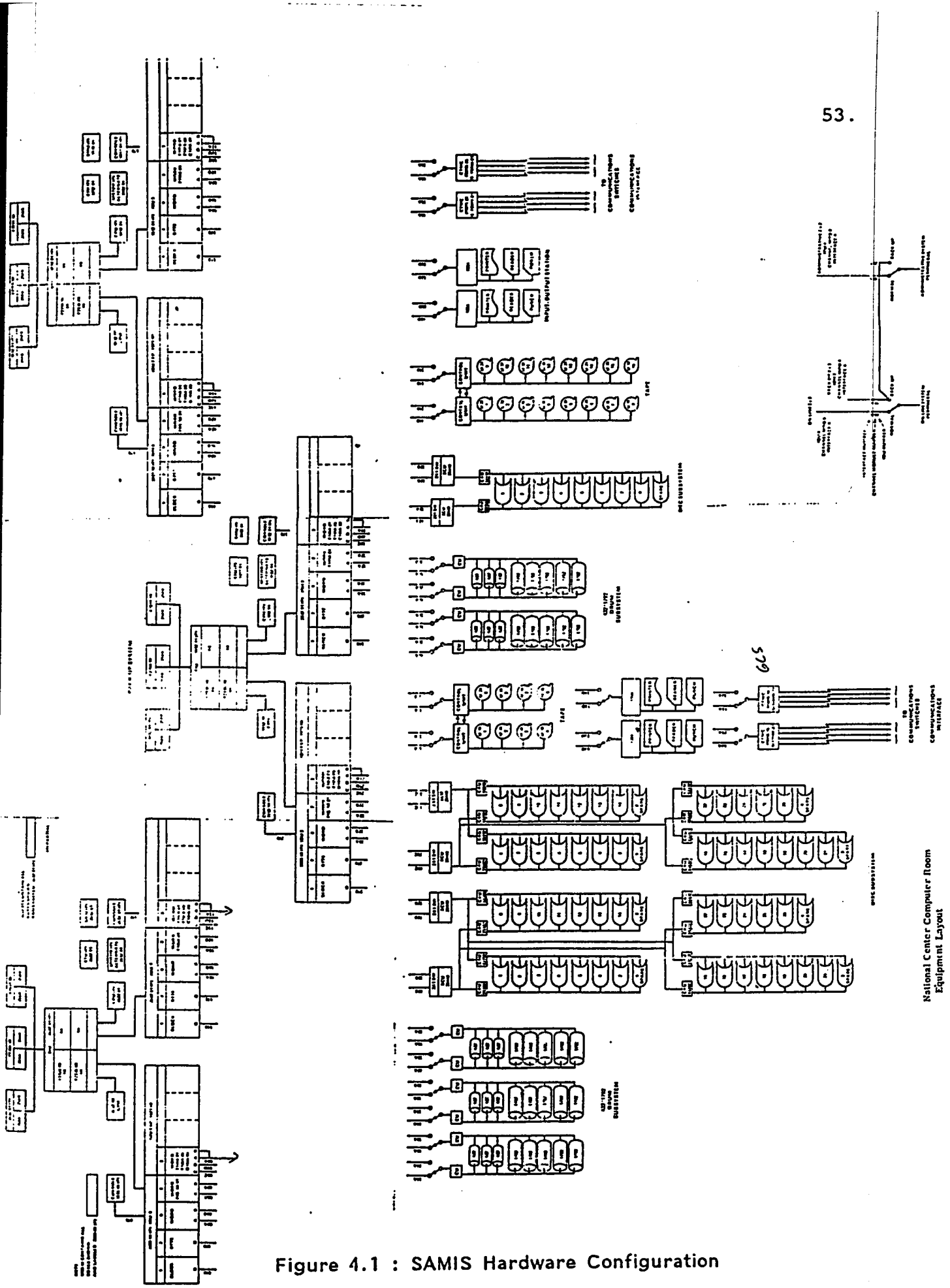


Figure 4.1 : SAMIS Hardware Configuration

CSTS-II Memory Map

Online System

All Addresses and sizes in Octal words.

1100/82 Configuration

34,000,000 - 37,777,777 Lower SIU - 1 MEG = 2048 pages
40,000,000 - 41,777,777 Upper SIU - 1/2 MEG = 1024 pages

Page size = 512 words
Word size = 36 bits, 9 bit bytes

Supervisor	Bank	Start	End	Size	Description
Coordinator	I-Bank	34,000,000	34,072,777	73,000	Kernel Code
Coordinator	I-Bank	34,073,000	34,137,777	45,000	Adjutant Code
Coordinator	D-Bank	34,140,000	34,142,777	3,000	Kernel Process Control Block
Coordinator	D-Bank	34,143,000	34,147,777	5,000	Kernel Data
Coordinator	D-Bank	34,150,000	34,172,777	23,000	Adjutant Data
Coordinator	D-Bank	34,173,000	34,246,777	54,000	Coordinator Services Scratch Space
Coord./Sys.	D-Bank	34,247,000	34,356,777	110,000	I/O Tables
User Swap Space		34,357,000	37,777,777	3,421,000 = 1809 pages	
User Swap Space		40,000,000	40,730,777	731,000 = 473 pages	
					2282 pages
System	I-Bank	40,731,000	41,115,777		System I-Bank
System	D-Bank	41,116,000	41,527,777		System D-Bank
System	D-Bank	41,530,000	41,536,777		Misc. D-Bank
		41,537,000	41,746,577		Misc. D-Bank
Coordinator	D-Bank	41,777,777			System Proc. Control Block

Figure 4.2.a : CSTS-II Memory Map

System D-Bank Break Down	Start	End	Description
System Tables	205,000	213,777	Service Work Block Initial Tables Shared Bank Tables
Communications Buffers	232,000	343,777	
List Space	344,000	441,777	OS Services Scratch Space
Page Management Buffers	442,000	567,777	Transient Code Disk Pages
Misc. D-Bank space	570,000	575,777	Task + File recovery data

Figure 4.2.b : CSTS-II Memory Map (continue)

Drum subsystems.

There are two types of drums, namely type 432 drums with 262,144 words capacity, 4.3 millisecond average access time, and type 1782 drums with 2,097,152 words capacity, 17 millisecond average access time. There are three drum strings, and each drum string has 3 units of 432 drums and 5 units of 1782 drums. Each drum string has two control units connected to seperate Input and Output Unit (IOU) word channel interfaces, therefore upto six concurrent drum input and output operations can be carried out. Current drum to interface connections are shown in table 4.2.

4.3 SAMIS Online System : System Components

The Operating System

The operating system CSTS-II is developed by the Computer Science Corporation (CSC) of Los Angeles, California. CSTS-II is a multisharing and multitasking operating system. The Interaction Management Protocol (IMP) is a CSTS-II subsystem designed to switch a large number of designated terminals among an integrated collection of executable programs. IMP is a facility provided within the CSTS-II operating system for performing transaction processing. [25]

Storage Control Unit	Drives	IOU	Channel	Interface
primary : 50	00-37	1	0	C0
secondary : 51	00-37	0	2	C0
primary : 53	40-77	1	0	C0
secondary : 52	40-77	0	2	C0

Table 4.1 : Disk to IOU Channel Interface Connection

String	Control Unit	Number	IOU	Channel	Interface
1	1	30 : secondary	0	1	C0
1	2	40 : primary	1	2	E0
2	1	31 : primary	1	1	C0
2	2	41 : secondary	0	1	E0
3	1	32 : secondary	1	1	E0
3	2	42 : primary	0	2	E0

Table 4.2 : Drum to IOU Channel Interface Connection

The Database Management System

MANAGE is the database management system used by SAMIS. It is a relational database mangement system, designed to provide a flexible, easy to use method of creating and using data bases on CSTS II. [23,24]

Files processed by CSTS II and MANAGE

MANAGE supports three types of data base file organizations namely, unkeyed, indexed sequential and hashed. These files are used within the SAMIS Online System environment [15]. Database file accesses are the major time consuming elements in the processing of information in the SAMIS Online system, so they should be very clearly understood and incorporated in the model.

The types and properties of the database file organizations supported by CSTS II Operating System can be summarized as follows.

1. UNKEYED files, have the folowing features :

- No primary key.
- File can only be processed sequentially.
- Once records are initially input, no more records can be added.

2. HASHED files, are used within SAMIS to reduce contention, particularly for small files. It is a

faster access method than the indexed organization. A hashed file is used when the following criteria is met :

- Value of the primary key is predictable.
- High volume of access is anticipated.
- No requirement for sequential processing exists.

3. INDEXED SEQUENTIAL files are the most expensive, since it requires many drum and disk input and output opeartions to access a specific record in the file.

CSTS-II Operating System references 5 types of files, as shown below [15].

UNP - Unpunctuated, a linear collection of words divided into physical records.

PNC - Punctuated, composed of logical records superimposed on physical records.

LIB - Library index, a collection of file descriptors.

REL - Relocatable,object code produced by a language processor.

XQT - Executable, absolute code produced by the linker.

LIB,REL,XQT file types will not be considered in the modeling procedure, since they are not used in any database accesses to actual files. They are system produced files, and could be considered as part of the system management techniques.

4.4. SAMIS Online System : Application System Components

The application system, custom developed by the CSC Corporation, covers the major parts of the computer processing requirements of the Saudi Arabian Ministry of Interior.

The Subsystems

The Online System is subdivided into eleven subsystems, called elements. Each handles a specific subset of information carried out by MOI. Miscreant, Passport Issuance, Criminal Records, Alien Control, Border Control, Drivers License, Motor Vehicle, Pilgrim Control, Civil Registry, Micrographics, and Message Communication subsystems presently constitute the Online System.

Terminals are usually hardwired to work on one of the elements of the SAMIS system. When the terminal is turned on, a menue for that element will appear, giving the operator the choice of several functions for that element. An example screen is shown in Fig. 4.3 [14].

SAMIS Hierarchy

SAMIS is functionally organised into the hierarchy shown in Fig. 4.4. The functional concept is the user's perspective of SAMIS. The SAMIS system is at the highest level, followed by the Security, Services, and the

Administrative subsystems in the second level. Each subsystem is organised into a set of application elements, such as Civil Registration, which are composed of a number of Functions, such as Register Marriage. Table 4.3 lists all functions in all elements of SAMIS Subsystems.

Functional Concept

A function is composed of additional entities called transactions, as shown in the SAMIS functional concept diagram in Fig. 4.5. A set of one or more transactions is required for each function. Each transaction is composed of one or more steps. A step is the smallest component of functional processing in SAMIS.

A function must begin with a transaction which must begin with a step. A function must stop at the end of a transaction which must stop at the end of a step.

Structural Concept

The structural concept is the programmers persepective of SAMIS. Fig. 4.6 shows a structural view of SAMIS within the host software. The major structural component is a program. Programs are composed of a main program and a number of subprograms, each of which are separately compiled, but all of which must be linked together and loaded as a task for execution. Each main program and subprogram is considered a module. A module is the smallest unit of code developed for SAMIS.

```
    0        1         2         3         4         5         6         7         8
    12345678901234567890123456789012345678901234567890123456789012345678901234567890
    ------------------------------------------------------------------------------
01! 9U01                    CIVIL AFFAIRS OFFICE FUNCTION                    09101!01
02!                                                                              !02
03! 01  REGIS HEAD OF                                                          !03
04! 02  REGIS DEPE                                                            !04
05! 03  ISSUE IDENTIFICAT                                                      !05
06! 04  ENCODE NATIONAL I                                                      !06
07! 05  FAMILY BOOK                                                           !07
08! 06  MARR                                                                  !08
09! 07  DIV                                                                   !09
10! 08  DEAT                                                                  !10
11! 09  IRA                                                                   !11
12! 10  UPDAT                                                                 !12
13! 11  QUERY                                                                 !13
14! 12  MICROGRAPHICS                                ENTER THE FUNCTI __      !14
15! 13  MESSAGE COMMU                                                         !15
16! 14  MISCREANT IDENTIF          ENTER THE NUMBER OF PAGES FOR MI  __       !16
17!                                                                           !17
18!                                                                           !18
19!                                                                           !19
20!                                                                           !20
21!                                                                           !21
22!                                                                           !22
23!                                                                           !23
24!                                                                           !24
    0--------1---------2---------3---------4---------5---------6---------7---------8
    12345678901234567890123456789012345678901234567890123456789012345678901234567890
```

Figure 4.3 : Sample Initial Menu Screen

Civil Registration Element

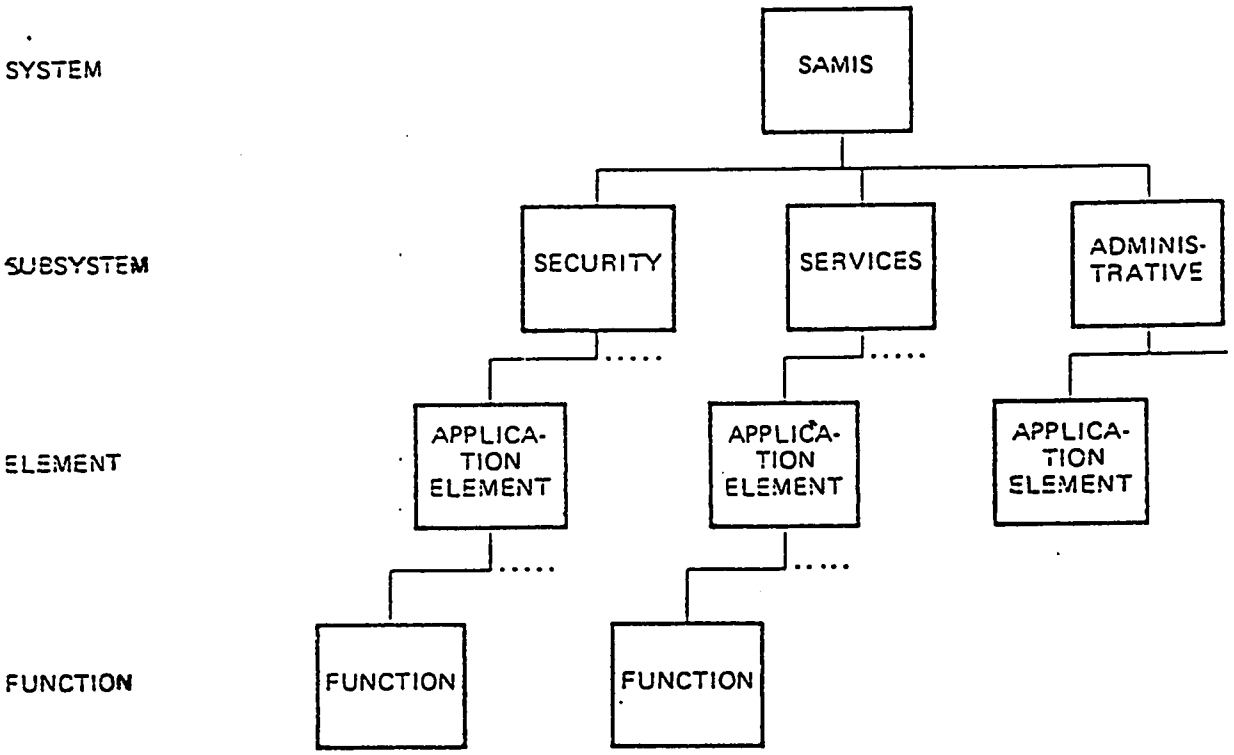


Figure 4.4. SAMIS Hierarchy

01-Miscreant Element
- 01-Request, Review and approval
- 02-New Miscreant Order Entry
- 03-Miscreant Order Revision
- 04-Temporary Miscreant Addition
- 05-Miscreant Deletion
- 06-Miscreant Encounter
- 07-Miscreant Report Generation
- 08-BlackList Query
- 09-Initial Blacklist Creation
- 10-Miscreant Check/Common Modules.

02-Passport Issuance Element
- 01-New Issuance, In-Kingdom
- 02-Reissue, In-Kingdom
- 03-Renewal, In-Kingdom
- 04-Out-of-Kingdom Issuance, Renewal,and Reissuance
- 05-Query Passport Records
- 06-Purge Passport Records
- 07-Generate Reports
- 08-Continue Miscreant Check.

03-Criminal Records Element
- 01-Record Criminal Event
- 02-Investigation Of Accused Person
- 03-Record Arrest
- 04-Record Judgement
- 05-Record Criminal History
- 06-Record Prisoner Detention
- 07-Record Prisoner Transfer
- 08-Record Prisoner Release
- 09-Name Clearance Request
- 10-Generate Reports
- 11-Criminal Record Inquiry
- 12-Revise Criminal Records
- 13-Delete Criminal Records
- 14-Criminal History Clearance
- 15-Generalized Reporting Capability.

04-Alien Control Element
- 01-Foreign Labor Authorization
- 02-Foreign Labor Authorization Change
- 03-Visitor Visa Extension
- 04-New Registration
- 05-Renew Registration
- 06-Exit/Re-Entry Visa
- 07-Final Visa
- 08-Sponsor Change
- 09-Lost Passport
- 10-Lost Passport Certificate
- 11-New Passport
- 12-Lost Alien Identification Card
- 13-Reports

Table 4.3.a List of SAMIS Functions

14-Query
15-Update
16-Cancel Visa
17-Travel Inquiry
18-Continue Miscreant Check.

05-Border Control Element
01-Saudi Arabian Citizen Entry Processing
02-Registered Alien Entry Processing
03-Visitor Entry Processing
04-Visitor Record Completion
05-Saudi Arabian Citizen Exit Processing
06-Registered Alien Exit Processing
07-Visitors Exit Processing
08-Travel Inquiry
09-Archiving Management and Statistical Reporting
10-Positive Miscreant Identification.

06-Drivers License Element
01-Drivers License Issuance
02-Drivers License Renewal
03-Accident Resord Processing
04-Traffic Violation Processing
05-Drivers License File Query
06-Revocation, Suspension, and Reinstatement Processing
07-Periodic Drivers License File Maintenance
08-Management Report Generation
09-Violation and Receipt Booklet Tracking
10-Violation and Receipt Booklet Resolution Control.

07-Motor Vehicle Element
01-Motor Vehicle Registration
02-Motor Vehicle Registration Renewal
03-Motor Vehicle Ownership Transfer
04-Motor Vehicle Query
05-Stolen/Recovered Vehicle
06-Registration Cancellation
07-Periodic Motor Vehicle File Maintenance
08-Motor Vehicle Management Report Generation
09-Motor Vehicle Mechanic File Maintenance
10-Temporary to Permanent Plate Issuance.

08-Pilgrim Control Element
01-Pilgrim Entry Processing
02-Pilgrim Miscreant Check
03-Pilgrim Detailed Identification
04-Pilgrim Exit Processing
05-Pilgrim Visa Extension
06-Pilgrim Exit Exception
07-Pilgrim Statistical Report Generation
08-Overdue Pilgrim Identification and Reporting
09-Pilgrim Query
10-Pilgrim Record Archival

Table 4.3.b List of SAMIS Functions (continue)

11-Positive Miscreant Identification.
09-Civil Registration Element
01-Citizenship Regaining
02-Naturalization
03-Registration
04-National Identification Card
05-Famil Book
06-Register Marriage
07-Register Divorce
08-Register Death
09-Withdrawn Nationality
10-Transfer Registration
11-Update Data
12-Change Name
13-Change Date of Birth
14-Query
15-Reports
16-Miscreant Identification.
10-Micrographics Element
01-Microfiche Assignment
02-Source Documents Transmittal
03-Source Documents Reciept
04-Micrographics Production Scheduling
05-Production Reconciliation
06-Source Documents Return
07-Production Summary Generation
08-Status Query
09-Micrographics Access For Criminal Element.
11-Message Communication Element
01-Mailbox Communication.

Table 4.3.c List of SAMIS Functions (continue)

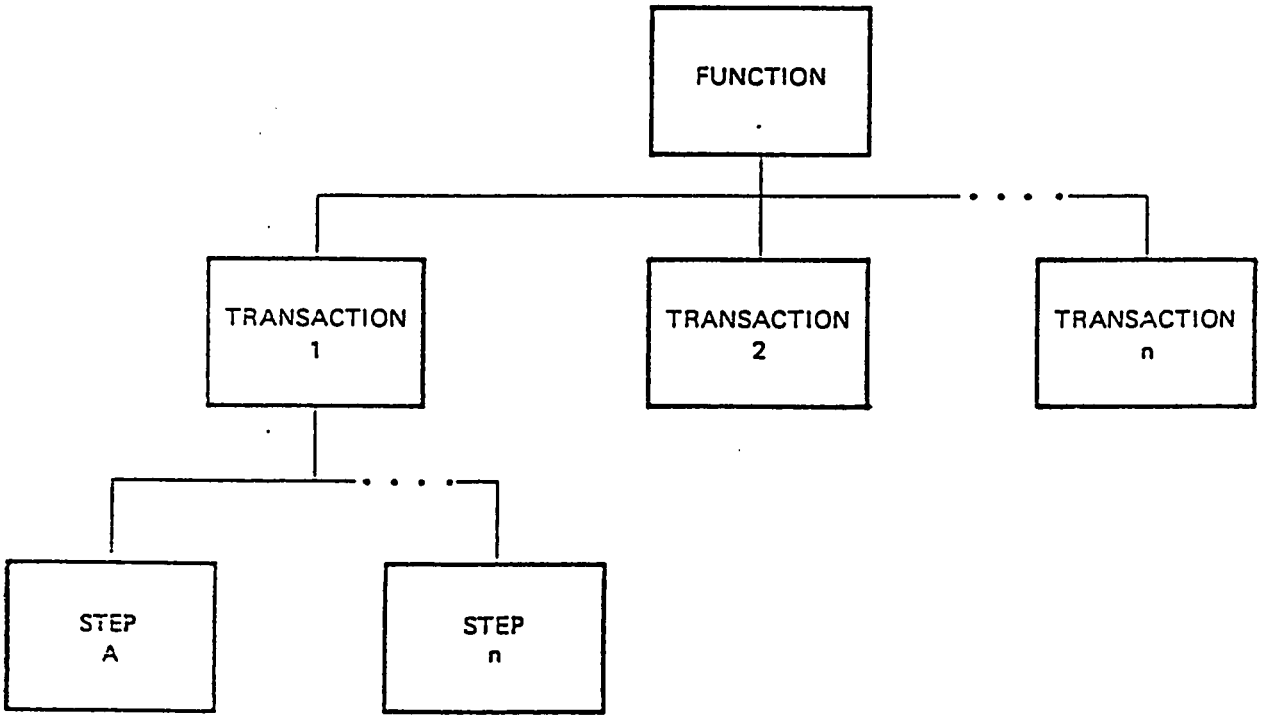


Figure 4.5. SAMIS Functional Concept

Fig. 4.7 shows an example step indicator chart (previously called a thread chart) which show the sequence and logic of module invokation for the Civil Registration element, Marriage function. The complete thread chart description of the SAMIS Online System is provided in [18,19,20,21]. From these charts the complete sequence of invokation of these modules is known the moment the function is invoked until it is complete.

The partial hierarchy chart in Fig. 4.8 shows main program to subprogram invokation and hierarchy. Fig. 4.9 and 4.10 show an example of module specification for a main program and a subroutine. In general the main program will perform the step processing required by the function. Subprograms provide system-wide services necessary for the main program's step processing.

Operational Concept

The operational concept is the programmer's persepective of SAMIS. It explains the order in which events occure to perform a functional unit.

The operation of receiving data from a terminal, processing the data, and sending an output screen to the terminal in reply is called an interaction. The interaction is the total amount of work performed by the host in response to a single user request.

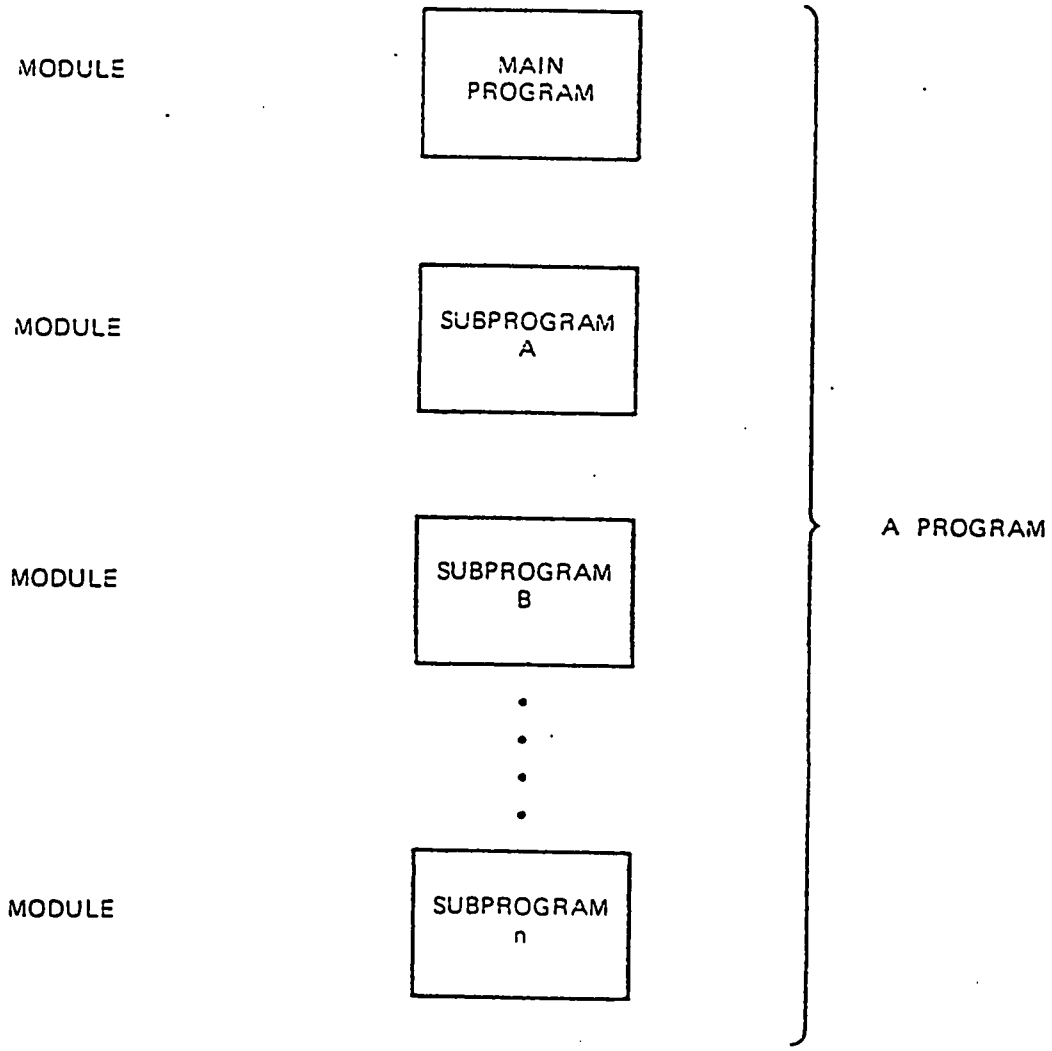
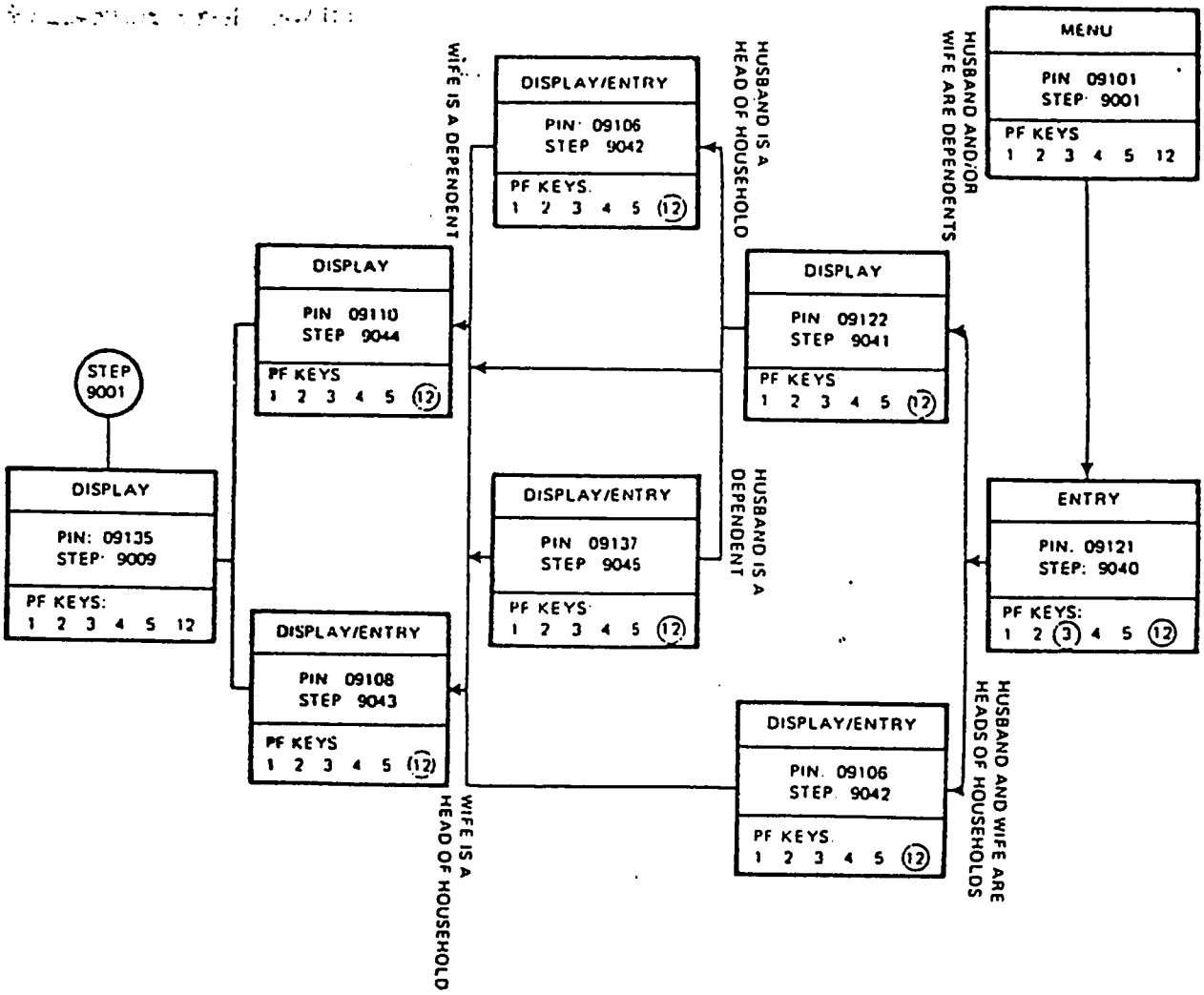


Figure 4.6. SAMIS Structural Concept



CIVIL REGISTRATION ELEMENT, MARRIAGE

Figure 4.7. Example Step Indicator Chart

(Thread Chart)

HIERARCHY CHART: REGISTER MARRIAGE

STEP CODE 9040

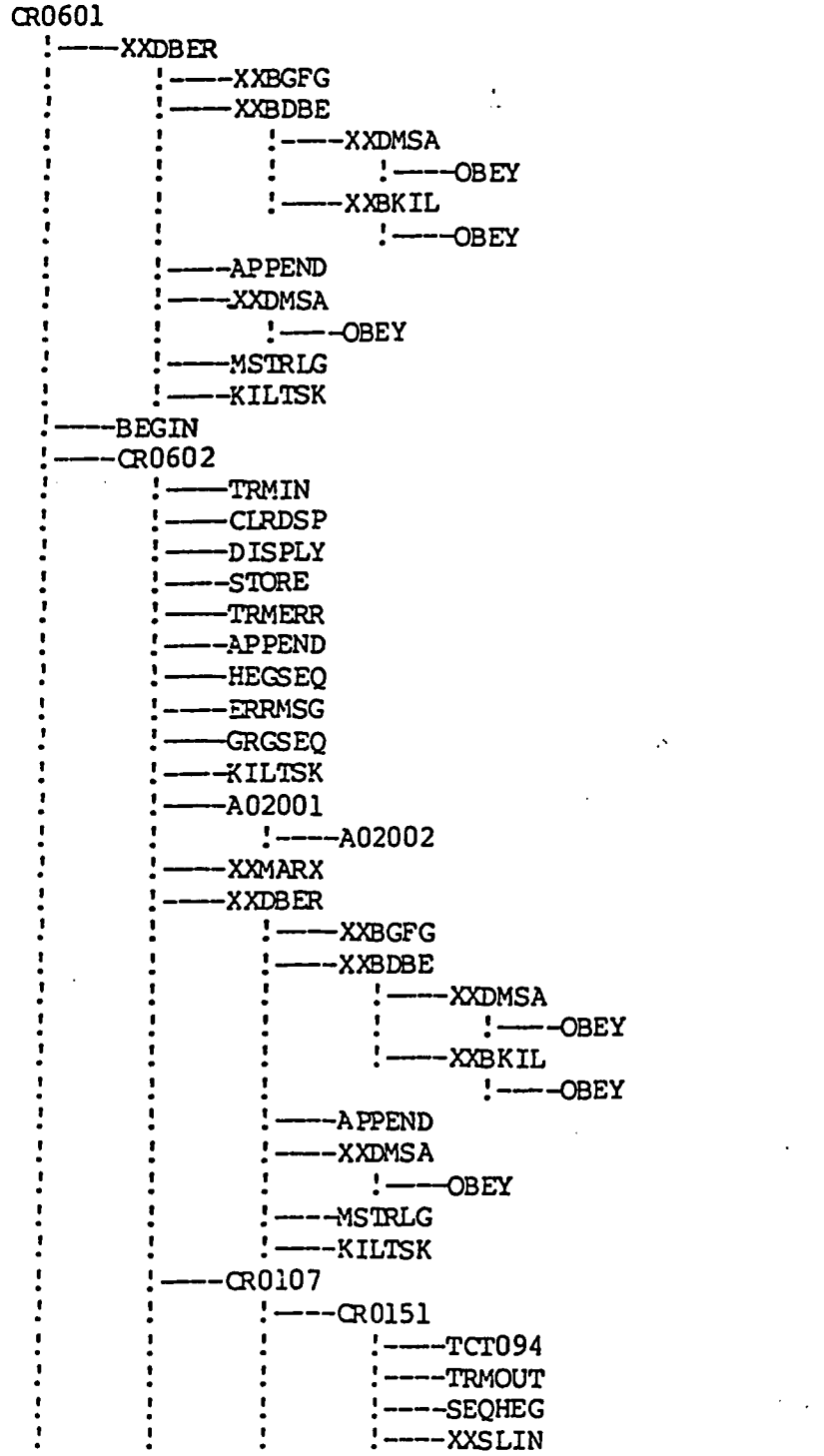


Figure 4.8. Example Partial Hierarchy Chart

MODULE SPECIFICATION

ELEMENT NAME:	CIVIL REGISTRATION	ELEMENT NUMBER:	09
MODULE NAME:	CR0601	MODULE TYPE:	MAIN PROGRAM
MODULE TITLE:	NONE		
MODULE PURPOSE:	DRIVER FOR STEP CODE 9040, WHICH VALIDATES ID NUMBERS AND DATE OF MARRIAGE		
ANALYST COMMENTS:	NONE		
CALLED MODULES:	BEGIN CR0602 KILSES XXDBER		
PIN NUMBERS:	09121		
SUBSCHEMA:	CV0301		
NON-MANAGE FILES:	NONE		
MODULE DESCRIPTION:	THIS IS A MAIN PROGRAM WHICH CALLS 'BEGIN' FOR THE STEP CODE 9040. IF THE STEP CODE IS INVALID, 'KILSES' IS CALLED. IF THE STEP CODE IS VALID, SUBROUTINE CR0602 IS CALLED TO VALIDATE NATIONAL ID NUMBERS OF THE HUSBAND AND WIFE AND TO VALIDATE THEIR DATE OF MARRIAGE.		
CALLING SEQUENCE:	NONE		
INPUT ARGUMENTS:	NONE		
OUTPUT ARGUMENTS:	NONE		
DATA DESCRIPTION:	NONE		
EDIT CRITERIA:	NONE		
DECISION TABLES:	NONE		

Figure 4.9. Example Module Specification - Main Program

MODULE SPECIFICATION

ELEMENT NAME: CIVIL REGISTRATION ELEMENT NUMBER: 09

MODULE NAME: CR0602 MODULE TYPE: SUBROUTINE

MODULE TITLE: NONE

MODULE PURPOSE: VALIDATES ID NUMBERS OF HUSBAND AND WIFE AND DATE OF MARRIAGE

ANALYST COMMENTS: NONE

CALLED MODULES:

A02001	APPEND	CLRDSP	CR0107	CR0151	TCT083	SEQGRG
DISPLY	ERRMSG	GRGSEQ	HEGSEQ	KILTSK	TCT088	AGECAL
NUMBCK	SEQHEG	STORE	TRMERR	TCT056	XXMARX	TCT027
TRMIN	TRMOUT	XXDBER	XXSLIN	TCT094	TCT006	

PIN NUMBERS: 09106 09121 09122 09137

SUBSCHEMA: CV0301

NON-MANAGE FILES: NONE

MODULE DESCRIPTION:

THIS SUBROUTINE EDITS THE DATE OF MARRIAGE AND THE NATIONAL ID NUMBERS OF THE HUSBAND AND WIFE FROM PIN 09121 FOR FORMAT AND CONTENT INPUT ERRORS. IT ALSO VERIFIES THAT THE HUSBAND AND WIFE EACH HAVE A CITZEN RECORD ON THE DATA BASE, AND CHECKS FOR POSSIBLE ERRORS REGARDING VERSION NUMBER, WITHDRAWN NATIONALITY, SEX CODE, OR PREVIOUS MARRIAGE.

FOR THE WIFE, IT CALLS CR0107 TO FORMAT PIN 09106 IF SHE IS A HEAD-OF-HOUSEHOLD, OR CALLS CR0151 TO FORMAT PIN 09110 IF SHE IS NOT. IF SHE IS CURRENTLY CARRIED ON A FAMILY BOOK OTHER THAN HER OWN, THEN THAT HEAD-OF-HOUSEHOLD IS CHECKED TO ENSURE THAT HE IS NOT A CURRENT HUSBAND OF HERS.

FOR THE HUSBAND, IT CALLS CR0107 TO FORMAT PIN 09106 IF HE IS ALREADY A HEAD-OF-HOUSEHOLD, OR FORMATS PIN 09137 IF HE IS NOT. IF HE IS CURRENTLY CARRIED ON A FAMILY BOOK OTHER THAN HIS OWN.

A FURTHER TEST IS DONE TO ENSURE THAT THE HUSBAND'S AND THE WIFE'S CURRENT HEAD-OF-HOUSEHOLD (IF ANY) IS NOT THE HUSBAND, OR THE WIFE, OR THE SPOUSE'S CURRENT HEAD-OF-HOUSEHOLD. THIS IS TO PREVENT FAMILY MEMBERS FROM MARRYING EACH OTHER.

Figure 4.10. Example Module Specification - Subroutine

Fig. 4.11 illustrates a sequence of events in the processing of a function. An interaction receives an input screen, processes the associated data as required, and sends an output screen, which in turn serves as the input screen for the next interaction. These screens are identified by a product identification number. The PIN number consists of five digits, the first two are the element identifier (Miscreant, Civil Registration,...), the third digit is a product type (screen, printed, microfilm...), fourth and fifth digits are unique numbers identifying different products (forms). For example PIN number (09120) means a screen display number (20) for the Civil Registration System (the Civil Registration System is element 9 in the list of SAMIS Online System subsystems). PIN number (09314) means a computer printout printed on a high speed printer, with code number (14) for the Civil Registration element. The screen with pin number (09101) has already been shown in Fig. 4.2.

It should be noted that the database files could be accessed at any time, but can only be updated at the end of a transaction.

The function in Fig. 4.11 consists of two transactions, the first transaction has two steps. Step 'A' receives a screen input from the operator, processes the output, may access the database, but may not update it.

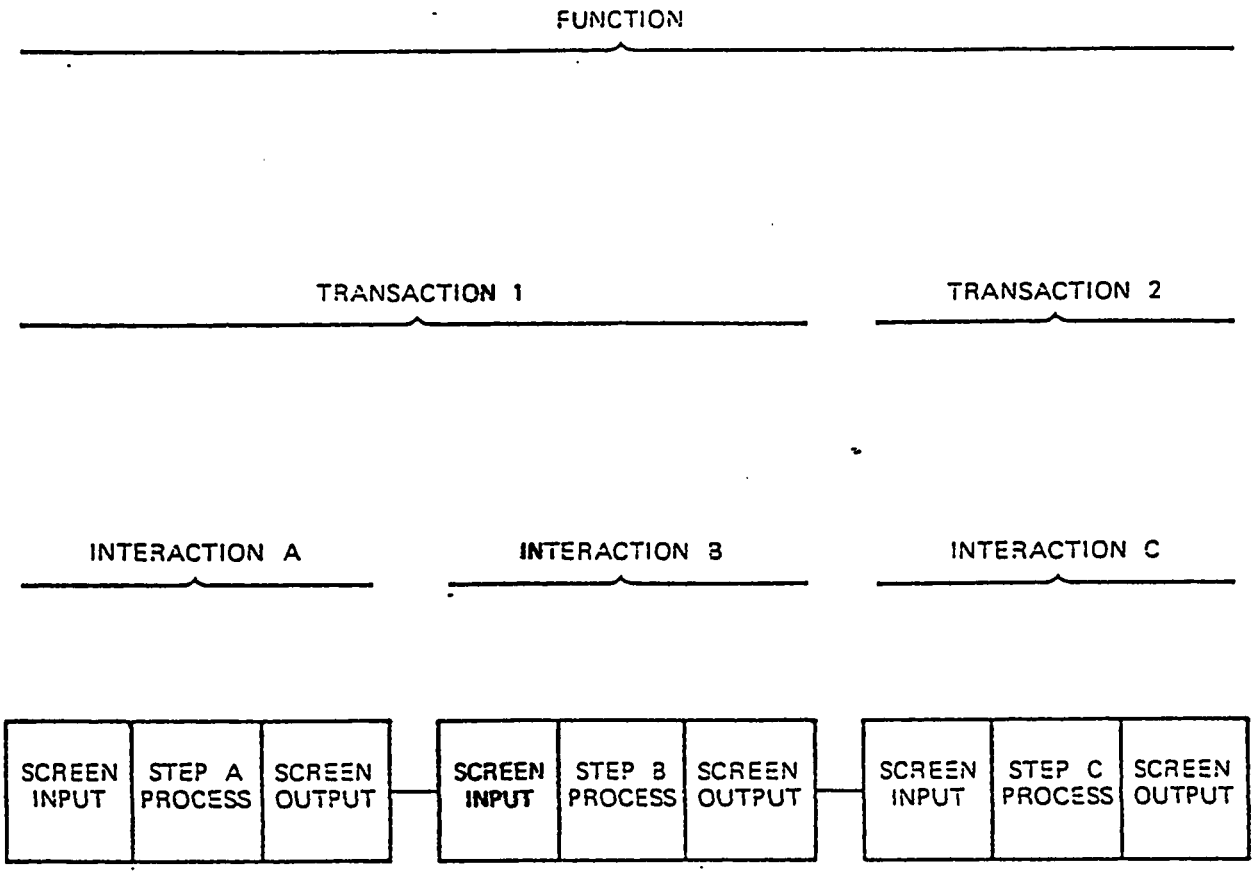


Figure 4.11 SAMIS Operational Concept

Step 'A' produces a screen output (second screen in this function) that is used as screen input by step 'B'. Step 'B' uses that screen input, processes the input, accessing and possibly updating the database, then producing the third screen output for this function. This is used as the screen input for step 'C', which is the only step in the second transaction. Step 'C' produces the fourth and last screen output for this function.

The execution of a module can be viewed as a sequence of scheduling, CPU bursts, memory allocation (for new task creation and swaped in tasks), drum and disk I/O cycles. The sequence is always followed by a terminal I/O which represents the display of information to the operator. Terminal I/O time includes the time for reply presentation, I/O delays, communication, delays and operator think time. These cycles depend heavily on the specific nature of individual functions, and are represented in the model.

SAMIS Tasks

When the screen data arrives at the host, the terminal is connected to the task that is associated with the stepcode of the current interaction. The task is executed and a screen output is produced to complete the current interaction. The execution of several such tasks in a sequence corresponds to the completion of one transaction of

a subsystem function.

A SAMIS task accesses several database files during its execution. Each file access requires some disk and drum I/O operations, depending on the file type. There may be several instances of a task at a given time, depending on the number of terminals being served. Task instances are under the direct control of the Interaction Management Protocol (IMP) which creates and deletes task instances as the load demands.

There exists a kill time that specifies the maximum time the tasks can wait without being invoked, after which the task will be released from the system. This is necessary due to limited main memory size, and the big number of tasks being requested by incoming transactions.

Communication between tasks is maintained through an Activity File (AF), which is placed on the drum workspace to speed up the processing [33]. The activity file is responsible for transporting data associated with operator terminals, from task to task, within a session, or between sessions. [27]

Functionwise Description

Functionwise description gives a more detailed understanding of the module behaviour. From this description details of file accesses, input and output of information, and frequency of file update could be obtained.

By combining the thread charts obtained from [21], and the functionwise description obtained from [14], it is possible to extract the details of each module within a function, specificaly, files accessed, how many times the files are accessed, and what input and output operations should be done.

As an example of a SAMIS Online System function, consider Fig. 4.7 where a thread chart of the Marriage Function of the Civil Registration element is shown. This function starts when the user requests this function from the screen displayed by the task whose stepcode is (9001), and the interaction commences. From this point on, execution will be done according to the sequence specified in the thread chart, and according to the input provided by the operator.

Stepcode (9001) is a menu which passes control to stepcode (9040), which is a data entry screen where the operator has to provide some input. Depending on that input, control will be either passed to stepcode (9041) or stepcode (9042). This procedure continues until stpecode (9009) is reached, where control will be routed back to stepcode (9001), and thus ending the interaction.

When control is passed to stepcode (9043), operations described in Fig. 4.12 will be carried out and files such as CITZEN, CIVHOH etc, are then accessed [14].

Description of all files and their properties are given in [15]. Fig. 4.13 gives an example of the properties of the (ALTRNS) file. Here the maximum record size, file organization and expected file size are clearly documented.

Given this information calculations can be made to compute the number of drum and disk input and output operations to be made when one access to this file is requested.

```
   TERMINAL OPERATOR:
      VERIFY INFORMATION ON PIN 09108;
      IF (INFORMATION ON PIN 09108 IS NOT CORRECT)
         CANCEL FUNCTION
      ELSE
         IF (PIN 09108 IS TO BE UPDATED WITH WIFE'S DATA)
            UPDATE APPROPRIATE FIELDS ON PIN 09108
            PRESS ENTER KEY
         END IF
      END IF
   END TERMINAL OPERATOR:
   SAMIS:
      REPEAT
         VALIDATE ENTRIES ON PIN 09108;
         IF (INPUT ERRORS EXIST) THEN
            DISPLAY ERROR MESSAGES ON PIN 09108;
            TERMINAL OPERATOR;
               CORRECTS ERRORS;
               PRESSES ENTER KEY;
            END TERMINAL OPERATOR:
         END IF
      UNTIL (NO INPUT ERRORS EXIST)
      UPDATE HUSBAND/WIFE/DEPENDENTS DATABASE RECORDS: CITZEN
      UPDATE HUSBAND/WIFE DATABASE RECORDS: CIVHOH
      PRINT HEAD OF HOUSEHOLD INFORMATION (PIN 09201)
         AND DEPENDENT INFORMATION (PIN 09202) ON COMTERM PRINTER;
      DISPLAY MICROGRAPHICS INFORMATION SCREEN
         (PIN 09135) WITH APPROPRIATE MICROFICHE INFORMATION;
   END SAMIS:
   TERMINAL OPERATOR:
      COPY MICROFICHE IDENTIFICATION AND FRAME NUMBERS
         ONTO APPLICATION FOR MARRIAGE (PIN 09405);
      ATTACH PRINTED FAMILY BOOK (PIN 09201 & 09202)
         ONTO APPLICATION FOR MARRIAGE (PIN 09405);
      PRESS ENTER KEY;
   END TERMINAL OPERATOR:
   SAMIS
      DISPLAY CIVIL AFFAIRS OFFICE FUNCTIONS MENU (PIN 09101);
   END SAMIS
END IF
END
```

Figure 4.12 : Sample PDL

Part of Marraige Function

2.3.4 Record Specification for Alien Transaction Record

NAME: ALTRNS (080)

MAX. SIZE: 9 words
36 bytes
ORG: Indexed
VOLUME (1990): 20,000

DESCRIPTION:

The Alien Transaction Record provides statistical data required to produce three monthly reports: Change of Sponsor Report (PIN 04311), Multiple Occurrence Lost/Stolen Passports or Alien Identification Cards (PIN 04307), and Lost/Stolen Passports or Alien Identification Cards (PIN 04313). The record contains the function code, location and date of the transaction, plus the identification number of the registered alien involved. The primary key of the ALTRNS record is the group element Alien Transaction Key (AL-TRANS-KEY), a concatenation of Alien Function Type (AL-FUNC-TYPE), Alien Transaction Place (AL-TRANS-PLACE), Alien Function Date (AL-FUNC-DATE), and Alien Identification Number (AL-ALIEN-ID). There are no alternate keys.

ALTRNS participates in no set relationships. Since the primary key is unpredictable, the record is organized as indexed. Its maximum length is 9 words (36 bytes). The Alien Transaction Record will be updated by the Change Sponsor Function, the Record Lost Passport Function, and the Replace Lost Alien Identification Card Function. The record will be purged once the monthly reports have been run. The expected volume is 20,000 records by 1990.

Figure 4.13 : Sample of File Description

File (ALTRNS)

5. THE SAMIS ONLINE SYSTEM MODEL

5.1. Abstraction

From the discussion in chapter two about model creation and the need to identify the parameters that actually affect the performance of the system, it is clear that the detailed study of the SAMIS Online System is neccesary.

From this study and according to the input obtained from experts on the SAMIS Online System, a number of decisions were made to include certain key elements and components as parameters of the SAMIS Online System in the attempt to build the model. These are not the only or the exact set of parameters that are expected to be incorporated for the creation of a SAMIS model, but they are the most appropriate set of parameters to be used, to the best of our knowledge.

More parameters could eventually be added to the model to enhance its resemblance to the actual system. Our method is hierarchical in the sense that we start by building a 'core' model, then expand the model by including more detail.

The important features of the components of the SAMIS system that have a direct effect on performance should be identified. Several assumptions were made when no clear understanding or no sufficent information

was available at the time the model was constructed. Most of these assumptions do not have a direct effect on the model structure and could easily be modified. These assumptions will be clearly identified in the text. Noting these assumptions is important for further use by future developers of this model.

The CSTS-II Operating System and the Manage Database Mangement System should be studied thoroughly in order to come up with the information needed for modeling. The Interaction Management Protocol [25] should be studied to understand the flow of transactions in the SAMIS system. From that the idea of transactions flowing in the model will be made clear. The study of the SAMIS Online System hardware components should then be conducted to construct the abstract of the SAMIS Online System. The study and abstraction of the SAMIS Online System software components represented by the operating system the database management system and the application system will then be conducted. The study of the hardware along with the software components of the SAMIS Online System will be merged together as a base of information to construct the model.

In the following discussion the abstraction of the software system components will be presented followed by the presentation of the abstraction of the hardware system components.

5.1.1. Subsystem , Function and Task Abstraction

The SAMIS Online Application System consists of eleven subsystems each of which is called an element of the SAMIS Online Application System. In turn these elements consist of functions, every element has a different number of functions depending on the definition of the element and what it actually does.

Each function within a subsystem consists of a sequence of tasks identified by a "stepcode". Invoking these tasks in a certain sequence produces a certain action which defines the function operation. The description of such a sequence in the model is important for the description of the function operation. The sequence of tasks invoked also depend on the input supplied by the operator.

Development of a methodology to determine the task sequence of each function and to represent it within the model is crucial to model the application system aspects of SAMIS Online System.

SAMIS Online System subsystems are given in [14] together with functions constituting each element. The program description language (PDL) providing the description of each of the functions is listed in [14] where the logical meaning of the execution of every function is decsribed.

The representation of the function description in a graphical manner could be found in [18,19,20,21], where a thread chart of each function in each element is shown. A thread chart describes the sequence of execution of modules or stepcodes in the functions. The thread chart does not however state the probability of transfer from one stepcode to another, due to data dependency on the operator input.

The PDL describes the logic behind the transfer, so the PDL should be studied, and the underlying probability of transfer should be assigned. This is not a trivial task however, since it is very much dependent on the understanding of the logic of each function in the SAMIS Online Application System. The basic information that should be extracted include the function identity number, the first stepcode to be executed when the function is invoked, and the probability of control transfer to subsequent stepcodes, listing the probability followed by the next stepcode. This information is repeated for all subsequent stepcodes, until the function description is over.

5.1.2. File Access Abstraction

Database access is another piece of important information about the stepcode that should be extracted from the PDL. A read or a write to a database file is considered an access to the database. A list of files

accessed by every stepcode along with the number of times the file is accessed by this stepcode should be extracted during the study of the function PDL.

At this point information about all elements of the SAMIS Online Application System, information about functions in every element, information about sequencing of stepcodes, probabilities of transfer from one stepcode to another in every function, and information about which files are accessed by every stepcode should be identified.

Thus the knowledge that is available currently enables us to know what stepcodes are to be executed when a certain function is invoked. It is also possible to know what files are to be accessed and for how many times, and when the execution of this stepcode is over. It is also possible to know according to the probabilities and sequencing information available what stepcode is to be excuted next, and when the function is to be terminated.

Knowing which files are to be accessed by a stepcode does not convey enough information, because it does not tell the number of disk and drum input and output operations that are going to be done while accessing these files. This would only be known if the characteristics of every file accessed by the SAMIS Online System database system are studied, and the average number disk and drum input and output operations that are done for each access to a record in these files are identified. This

information is extracted from [15] where all the files accessed by every function is identified and properties and characteristics of these files are described.

The maximum record size in the file, type of the file, key size if any, and the expected file volume for 1990 need be known. From this information the number of disk and drum input and output operations could be identified. The basic information that should be extracted include:

i. File name, which is used to identify the file when stepcodes of different functions try to access this file.

ii. Maximum Record Size, both in words and in bytes.

iii. File Type, is important to determine the number of drum and disk I/O operations to be made to access the file.

For an indexed file, the indices are stored in a key-index tree. Our interest is to know how the tree is traversed, to reach the index for a specific record of a specific indexed file.

The tree has a maximum of four levels, and every level implies one drum I/O operation. So if it was neccessary to go through the tree three

times to get a certain key to a file, (key on level 3 of the tree), then we need three drum accesses to get the index, since indices are stored on drum, and one disk access for the record itself, since the actual file resides on disk.

The depth of the key index tree actually gives the number of drum input and output operations needed to access an indexed sequential file and can be computed as follows [29].

$$NL = NUI(\log(NP)/\log(kp))$$

Where NL is the Number of Levels, NUI is the Next Upper Integer and NP is the Number of Pages. Number of pages is the file size in pages. A page is 512 words, and the size of the file is calculated by multiplying the record size (in words) by the expected volume of the file. File volumes in words are predicted for 1990 and listed for all files of the Online System in Appendix A [15]. File size is divided by 512 to get the anticipated file size in pages. KP is the number of keys per key page. This is calculated using the formula :

$$KP\ (\ Keys/Keypage\ ) = INTEGER\ [509/Keysize]$$

Keysize is in words, and for a keysize of length 1 for example, we can have 509 keys per one

key page. Keysize for each file can be obtained from [14] by examining the PDL description of the file.

iv. Volume, anticipated for the file in 1990 is used for calculating the number of drum accesses, that should be made to obtain the key of the indexed file.

v. Drum Number, on which the index tree of a specific file resides. It was decided to incorporate this specific association of drums to files, and not to randomize drum accesses according to availibility. This decision is due to the fact that some files might act as bottlenecks, when many functions try to access the same file, resulting in a busy drum and a queue of waiting transactions.

vi. Keysize, in words and is used in the calculation of the number of drum accesses.

At this point it would be possible to know the sequence of execution and invokation of stepcodes whenever the operator issues a request on any of the functions. With this the abstraction of the software element of the SAMIS Online System would be complete.

5.1.3. Hardware Abstraction

In order to make a hardware abstraction of the SAMIS Online System, the hardware components of the SAMIS Online System should be studied, and relevant components that appear to have an effect on the system operation and performance are identified and incorporated in the model.

The dual CPU units of the UNIVAC 1100/82 that run the SAMIS Online System should, of course, be incorporated in the abstract of the SAMIS Online System. The speed of the CPUs and the scheduling algorithms used should be defined. It is not a trivial task to represent the scheduling and service algorithms in the model, so after the study of the CPU structure and service criteria, it was decided to simplify the CPU scheduling to be first come first serve, without the inclusion of any priority handling.

There are actually two types of memory represented in the SAMIS Online system, the instruction bank of memory, called the I-Bank memory, and the data bank of memory, called the D-Bank. These two banks are assumed to share the available memory locations. The map of the main memory for the CSTS II operating system was shown in Fig. 4.2. The size of the user area in I-Bank is 1809 pages, and in the D-Bank is 473 pages, resulting in the total user

area size of 2282 pages, where a page is 512 words and a word is 4 bytes, resulting in a user area memory of 4.5 Mega bytes. Memory will be represented in the model as one type only. No special memory types are going to be represented in the abstract.

Disk subsystems and detailed information about disk scheduling criteria and input and output channels required to gain access to the disks are also included in the abstract. Features include minimum and maximum drive seek times, average latency, and page capacity.

Drum subsystem is included in the abstract along with the input and output channels required to obtain access to the drums. Characteristics including number of drums in a string, word capacity and average access times are also abstracted.

Inorder for a transaction to carry out disk or drum input and output operations, it should acquire a channel through which the disk or drum could be accessed. Drum and disk allocation are done according to the path the transaction is routed through the channels and the input/output units, drum and disk to channel connections were shown in Tables 4.1 and 4.2. There are 24 drums grouped in three strings. Drums are of two types, the FH-432 drum has an average access time of 4.25 milliseconds with 262,144 words capacity, and the FH-1782 drum with an average access time of 17.0 milliseconds with

2,097,152 words capacity. There are 64 CDC-33332 disk drives with a minimum seek time of 10.0 milliseconds, an average seek time of 30.0 milliseconds, a maximum seek time of 55 milliseconds, and an average latency of 8.3 milliseconds, and 78,090 pages capacity.

The details of the communication hardware and software are not considered in the abstraction of the SAMIS Online System hardware components. The only appearance of the communication system and operator think time is a specific delay representing the occurance of communication. More detailed incorporation of the communication system could later be added to the model.

5.1.4. Development of an IPG

In the modeling process using the PAWS computer simulation language (refer to figure 3.1) an Information Processing Graph (IPG) of the SAMIS Online System should be constructed representing the model and describing the information processing procedures conducted by the SAMIS Online System. A PAWS program will be constructed to translate the IPG to a computer processable model. A major component of this IPG and PAWS program is the user node, which is a FORTRAN 77 module written to incorporate all the features of the SAMIS Online System in the model's behaviour. This will be done by conveying all the features that were just described about the SAMIS Online System. The user node is the actual

interface between the PAWS model and all the information provided about the SAMIS Online System.

In constructing the IPG the flow of control should be shown along with information about resource management of the SAMIS Online System. To construct the IPG the following procedure should be followed.

First all active nodes, token types, and memories should be identified. Then the number of tokens of each type and the memory capacity at initiation should be specified. Following that different workload categories, network nodes, and connecting edges are identified. Finally the transaction routing behaviour is described.

Following this construction methodology we first start by identifying all active nodes, token types and memories. Active nodes include the CPU (dual), disk subsystems of 64 disk drives, and drum subsystems with 24 drum drives. One type of memory exists. Disk and drum input and output channels are considered as tokens.

There is a total of four types of drum channels and input/output unit configurations allowing upto six drum interfaces. Channel one on input/output unit zero has two drum input/output intefaces, and two interfaces on input/output unit one. Channel two on input/output unit zero has one drum input/output interface, and one drum

input/output interface on input/output unit one.

There is a total of four types of channel-input/output unit setups allowing for up to four disk interfaces, namely input/output unit zero with channel zero, input/output unit zero with channel two, input/output unit one with channel zero and input/output unit one with channel two. Each setup allowing one disk input/output interface.

The service assignment for disk input and output requests through these channel-input/output unit setups is identified and incorporated in the IPG.

Memory capacity will be initialized at run time and can be changed for different simulation runs to analyze its effect on the performance.

Transactions entering the system will be assigned categories according to the element (subsystem) numbers. Thus the source of transactions should produce transactions with eleven different types of categories, namely ALIEN, BORDER, CIVIL, CRIM, DRIVE, MISCR, MOTOR, PASS, PLGRM, MICRO AND MSG. Each category actually represents one of the eleven SAMIS Online subsystems.

Transactions entering the systems should be treated equally in terms of file accesses, CPU scheduling and so forth. So the transaction category will help only in the collection of statistics on different elements of the model, but will not play a major role in transaction

routing through the model as the case might be in normal PAWS models.

Routing the transactions will actually be done through the use of the phase of the transaction rather than the category. Phase control will be mainly done according to specific conditions arising due to some situations in the status of the model, and will be made usually in the special user node.

The user node is a special node that is designed to incorporate special features to be implemented in the model. All requests done by the operator, which are usually function invokation, and represented in the model by transactions entering the model, are routed in the model to the user node where it identifies the request and associates it with a certain known function in the model. The sequence of stepcode invokation is then initiated according to the function description represented in the user node.

A special processing section defines actions taken when certain transactions flow in the system. These transactions would be coming mainly from the user node, reflecting certain conditions implied by the SAMIS Online System model.

Not all transactions flowing in the system coming from the user node request service of one of the resources, or request acquisition of memory units or tokens. Some transactions are control transactions that initiate some

special processing such as swaping in a swaped out task, waking up a dormant transaction, and so forth.

5.2. Basic IPG and PAWS Program

Based on the abstraction described in section 5.1, the basic IPG shown on Fig. 5.1 has been constructed. A detailed discussion of the IPG is given in section 5.3 and 5.4. From this IPG, a PAWS program has been written to translate the model to a machine understandable program. This program would then be run several times to obtain required statistics for convenient analysis of the SAMIS Online System model.

The current model is heavely dependent on the user node. Most of the processing is done inside the user node and PAWS is used as a tool for statistics collection and some commands issued by the user node.

Having this basic IPG it would be possible to construct an initial PAWS program that would be able to represent the properties of the SAMIS Online System hardware abstract. Presenting the SAMIS Online System software abstract is going to be done inside the user node. It would not be possible to explain the details of the special processing section of the IPG unless the processing details of the user node and its effect on transaction phase numbers is made clear. Thus the discussion of the detailed IPG will be done when the complete explanation of the user node is over.

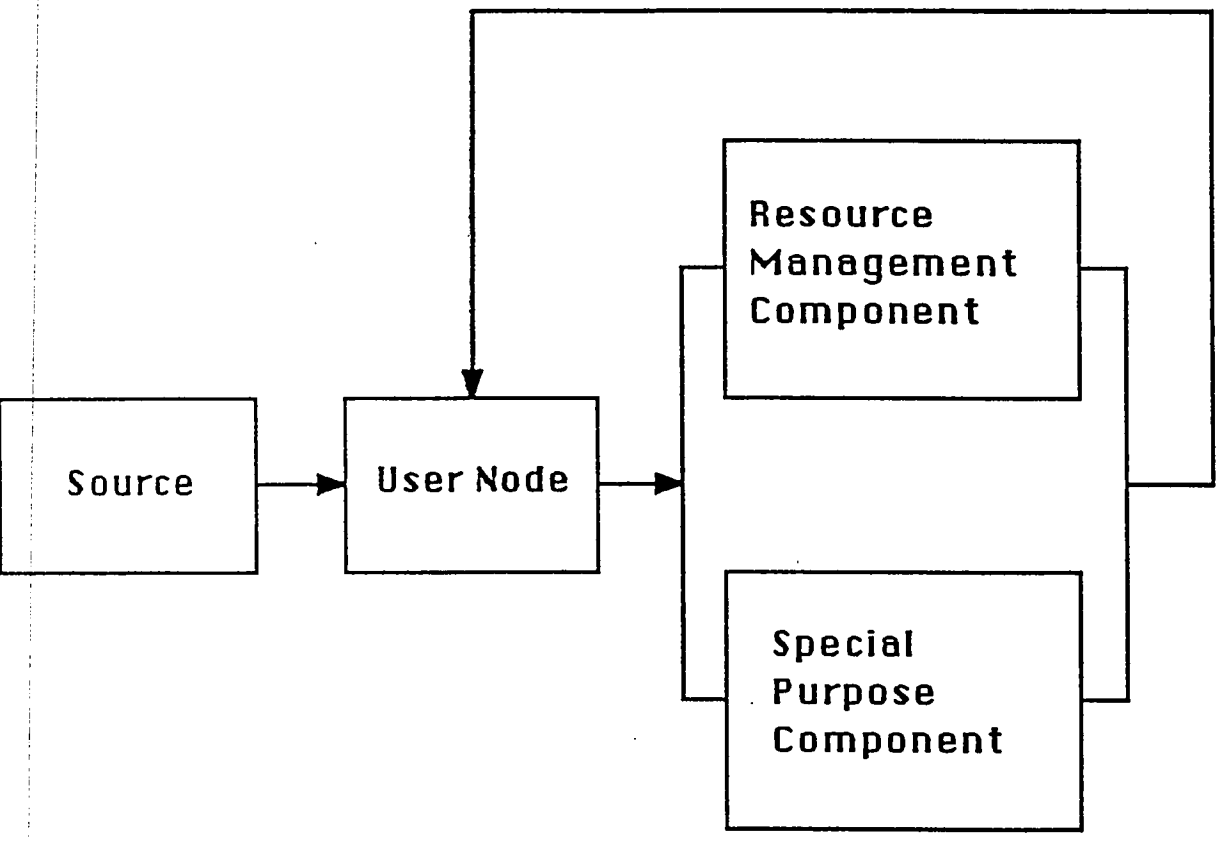


Figure 5.1 : Basic Information Processing Graph (IPG)

Translating the basic IPG to a PAWS model using the PAWS language is a straight forward procedure. The IPG shows the interconnection of the nodes in the model. The PAWS program has a more detailed definition of the nodes in the model.

The DEFINE section in the PAWS program needs special attention because that is where the characteristics of nodes in the model appear. The main argument in the remainder of this section will be on the DEFINE section of the PAWS program, and the explanation of the details of the user node.

Starting with the source node (E) it has been decided that transactions entering the system have arrival times that are exponentially distributed, having a mean that is being set as a parameter for studying the system performance. Smaller mean arrival time implies a heavier system load. This parameter will be changed during the experimentation phase to study the effect on the overall system performance. The source node supplies transactions with eleven different categories, namely ALIEN, BORDER, CIVIL, CRIM, DRIVE, MISCR, MOTOR, PASS, PLGRM, MICRO, and MSG, each representing an arrival of function request by an operator for that function. Fig. 5.2 Shows the details of the source component of the basic IPG.

The dual CPU single scheduling algorithm results in a

PAWS node with a single input queue and dual servers. The queueing discipline for the CPU units is assumed to be first come first serve. CPU quantum time is assumed to be exponentially distributed with a mean that is set as a system performance parameter.

Memory will be considered as one type only in the SAMIS Online System model. No I-Bank D-Bank distinction will be implied in the model, and the two memory types are assumed to share the available memory locations. Memory size will be assumed 2282 units, and a unit is one page.

When a task requests memory, I-bank and D-bank memory sizes need to be specified. These amount are going to be passed to the memory management node, where all requesting transactions are going to be serviced according to the first come first serve scheduling criteria, and the memory will be allocated if available according to the bestfit memory management technique.

When memory is being allocated to a certain transaction, it will be supplied also with the addresses of the memory locations of the allocated memory assigned to it. This information will be used when a memory release operation is initiated. Memory release is done in the RMEM node where the memory bank allocated to the transaction with the specifed address will be released and the memory is made available for use by other transactions. Task memory requests are dependent on the

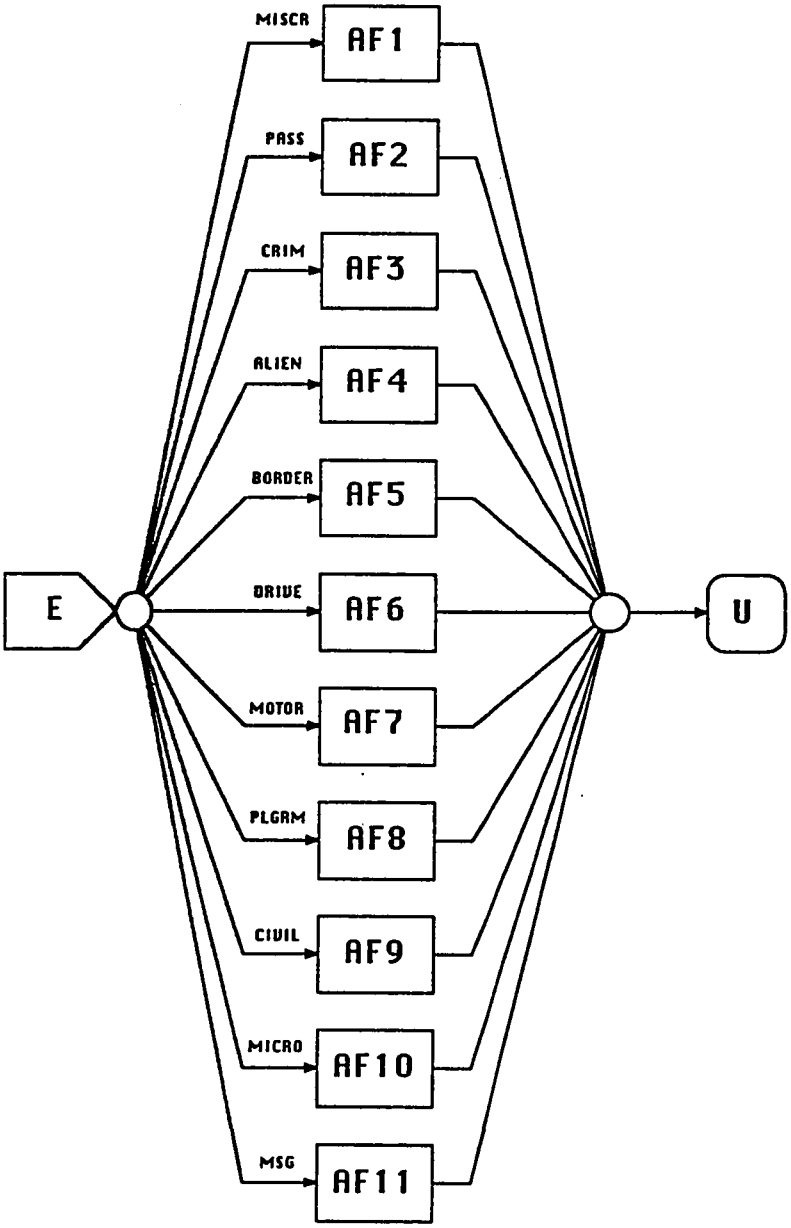


Figure 5.2 : Details of the Source Component of the Basic IPG

specific task. Complete information was not available when the model was constructed. Partial information on the I-bank and D-bank memory sizes was available for some of the tasks. The sum of the I-bank and D-bank was available for all tasks, thus a weighted average was computed representing the I to D bank ratio using the information about the available tasks, and reflected on other tasks of unknown I-bank and D-bank sizes, thus producing an estimate of these requirements for all tasks. The reason for this procedure is that actual I and D bank memory sizes are not fully available from NIC.

In order for a transaction to carry out disk or drum input and output operations, it should acquire a channel through which the disk or drum could be accessed. Scince the channel is something that should be held by the transaction inorder to obtain service from the disk or drum, and it does not itself provide any service, and it will be held for the duration of disk or drum input and output operations, then the channel should be considered as a token that should be acquired before getting service from the diks or drums, and should be defined to be so in the PAWS program.

Drum and disk allocations are done according to the path the transaction is routed through the channels and the input/output units. There are 24 drums grouped in three strings. Drums are of two types, the FH-432 drum has an average access time of 4.25

milliseconds, and the FH-1782 drum with an average access time of 17.0 milliseconds. Drum service time is approximated to be simple delay equivalent to the average access time. Data transafer time is ignored considering the speed of the drums. More precise definition of drum access times may be included in the model as empirical data. Drums are allocated to transactions based on requests issued from the user node, and are allocated on a first come first serve basis.

Transactions are randomly routed to one of the 64 CDC-3332 disk drives since the current NIC policy is to distribute all Online system files on disk drives as uniformly as possible.

Disks are allocated according to a first come first serve scheduling criteria and have a uniformly distibuted service time between 18.3 and 63.3 milliseconds. (minimum seek time + latency, to maximum seek time + latency). Disk transfer rate is ignored.

One very important assumption that has been made regarding the disk controllers is that the controllers remain connected to the disk drive while the drive is carrying out the seek operation. This has a direct effect on the performance of the disk drives. In addition all channels (tokens) should be released after the service is done so that other transactions could use these facilities.

Details of the resource management component of the basic IPG is decribed in Fig. 5.3. Fig. 5.4 shows the details of the drum-channel connections, and Fig. 5.5 shows the disk-channel connections.

Other features of the IPG could only be described when the transaction flow in and out of the user node is explained.

5.3. User Node

Transactions entering the PAWS model simulate actual requests by the SAMIS Online System operators for the invokation of certain functions. When transactions are initially created in the source node of the PAWS model and enter the system, they will be directly routed to the user node, which is a FORTRAN 77 module written to incorporate all the features of the SAMIS Online System. The user node will then check if these transactions are known to the model or not. The user node keeps some data structures to identify all transactions routed to it.

5.3.1. Data Structures

Following the abstraction process described in section 5.1, the information about all function identity numbers of the SAMIS Online System (e.g. function 0906 is Register Marriage Function) have been placed in a table called the function pointer table (FP).

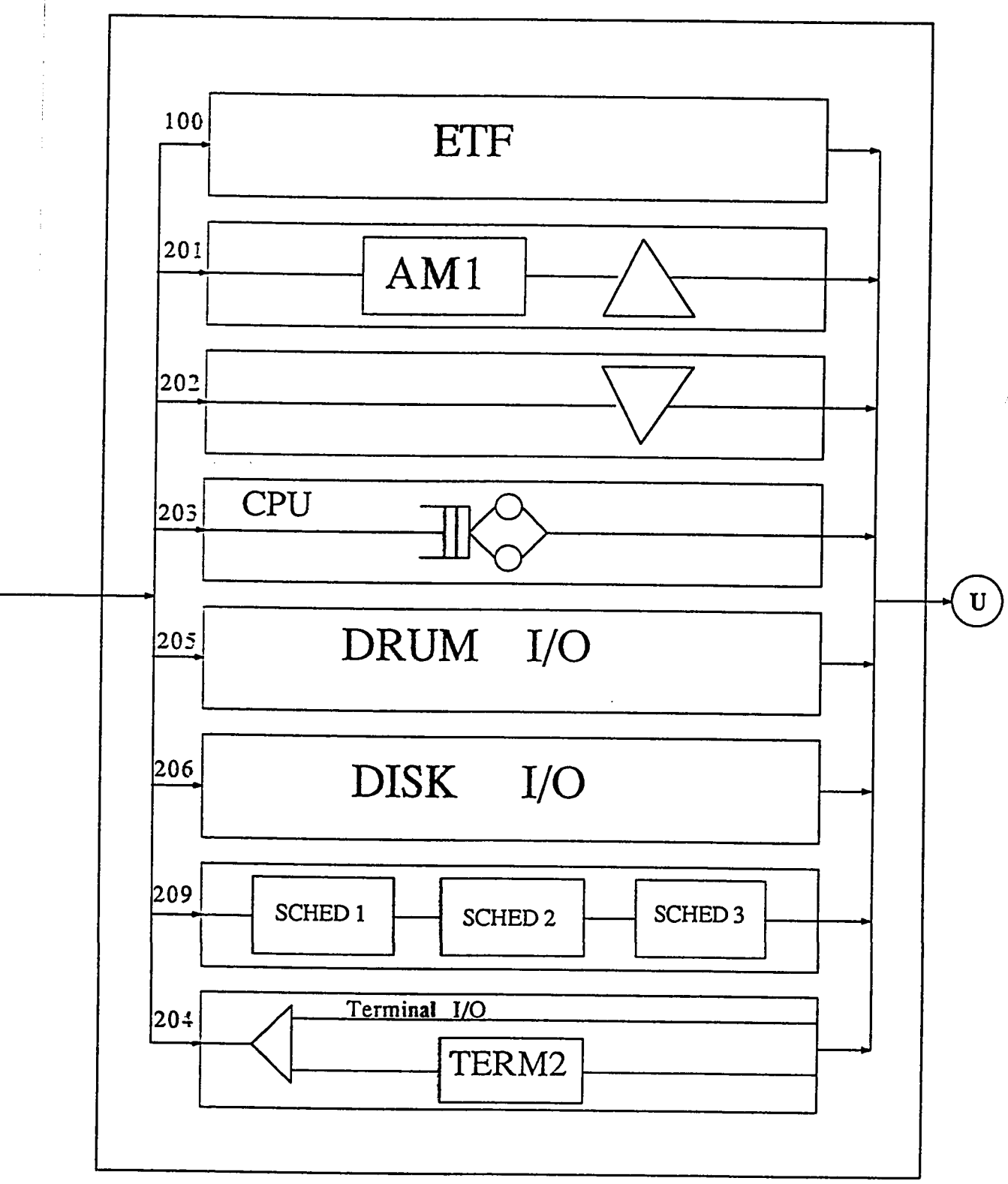


Figure 5.3 : Details of the Resource Management Components
Basic IPG

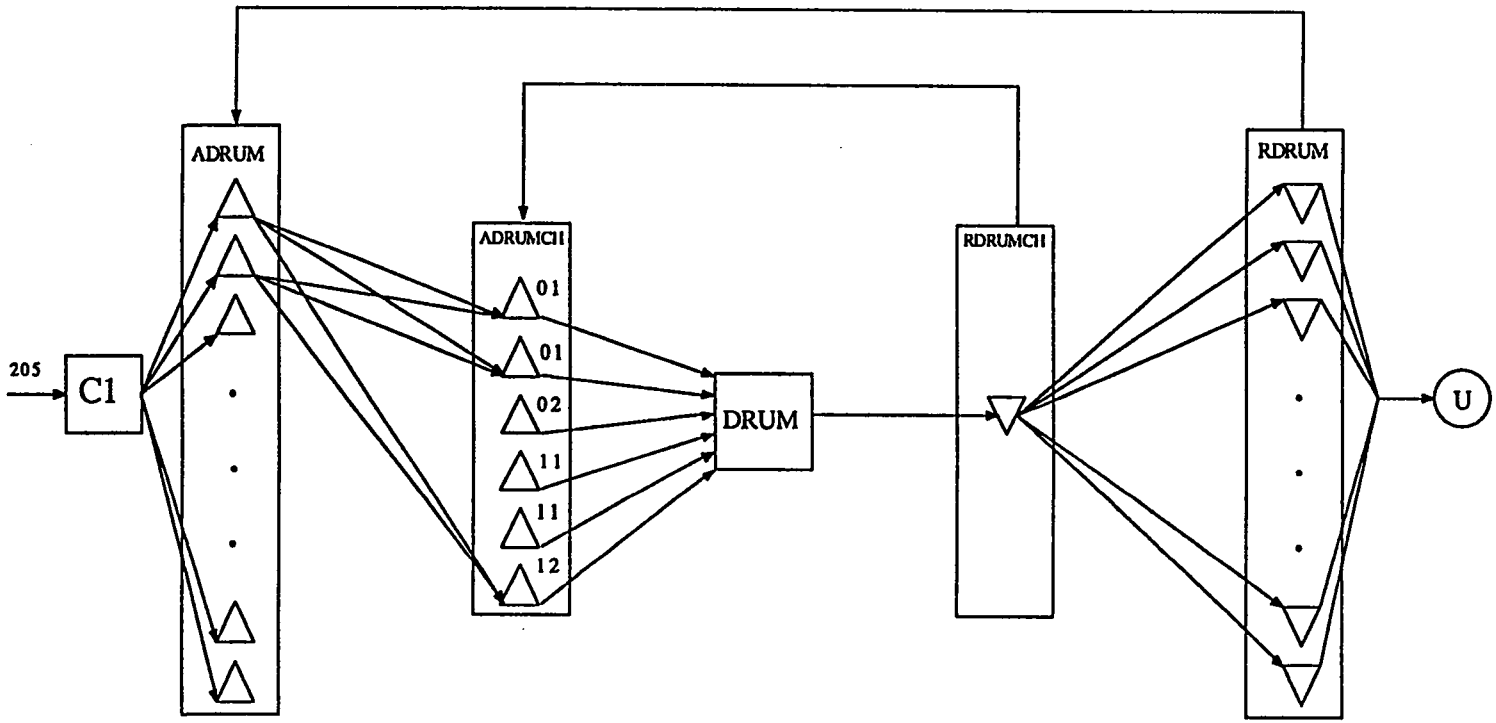


Figure 5.4 : Details of Drum/Channel Connections
Resource Management Component
Basic IPG

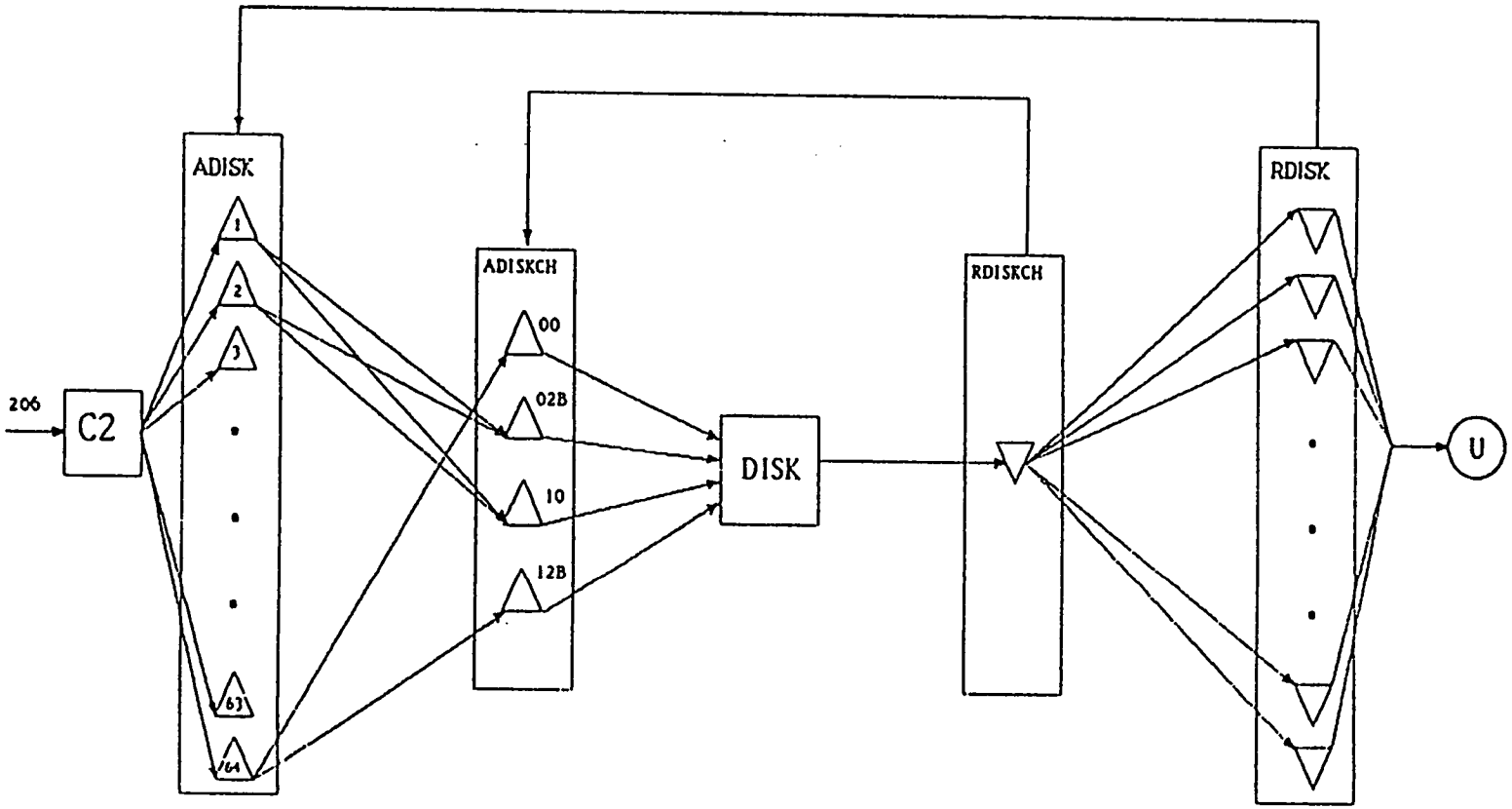


Figure 5.5 : Details of Disk/Channel Connections
Resource Management Component
Basic IPG

Functions are composed of stepcodes whose sequence of execution actually describes the actions carried out by the function. Information about the structure of every SAMIS Online System function was extracted and placed in a data structure that is called the function sequence table (FS). FS table includes information about every function interms of the sequence of invokation of stepcodes in that function, probabilities of control transfer from one stepcode to another, and when the function processing is over.

An entry in the FP table identifies the first stepcode that is to be executed by the function when it is invoked. The user node will look for the function identity number of the function in the FP table, then if found it will identify the first stepcode the function will execute through this pointer to the FS table.

Every stepcode that is described in the SAMIS Online System will have what is called a "task" that will be associated with it. This task will be routed through the model representing the execution of the stepcode. There could be more than one task copy representing this stepcode, to ensure better service and enable concurrent satisfaction of user requests.

Table FS has an entry that would point to some other data structures having detailed information about the

task copies associated with the stepcode. These data structures consist of twenty one tables with different dimensions. All of these tables will be collectively called the "Task Information Table" (T).

Table T is composed of two major types, namely data tables representing the properties of the stepcode the task is associated with, and tables used for the manipulation and management of the information obtained during the simulation time representing the status of the task copies simulating elapse of actual time periods in the SAMIS Online System.

In more detail, table T contains information about the task identity number, the task stepcode number that this task is associated with, and the I-bank and D-bank memory requirements of the task.

Task copies including the original task copy, are created on request by the user. A special flag is set if a task copy is to be created at system startup. This might be necessary for important tasks such as menus that should be created and available before any user requests their function.

It is possible to create copies of the task depending on execution demand. A copy is created if a certain number of users request the service of this task, and the task copy is busy serving some other user.

The number of users waiting after which another task copy could be created is provided as one of the entries in the task information table. A maximum number of allowable copies for each task is specified in table T.

After task copies are created, if a certain period of time passes without any request arriving for task service, then the copy is taken out of the system. The kill time for task copies is speified along with the kill time for the original task copy.

The number of task copies existing in the system along with their states and memory locations assigned to them is also kept in table T. The count and identity numbers of waiting transactions for service of each task are also saved in table T.

A special entry in the table is the information about all files that are accessed by the task. This entry has been constructed by a special utility program that uses information passed to it about the stepcodes and files they access.

Table 5.1 shows a summary of the definition of the FS and FP table, and Table 5.2 shows a summary of the definition of data strucrtures (T). Figure 5.6 shows a pictorial description of relations between these data structures.

TABLE FP : FUNCTION POINTER TABLE

TABLE FS : FUNCTION SEQUENCE STEP INDICATOR TABLE

FP1 : FUNCTION ID, XXYY

FP2 : POINTER TO THE FIRST TASK OF THIS FUNCTION IN FS

FS1 : STEPCODE OF A FUNCTION TASK. IF IT IS ZERO THE TASK STEPCODE IS THE SAME AS THE ONE THAT HAS THE CLOSEST LOWER TASK INDEX. THIS IS USE TO DEFINE BRANCHES IN THE TASK SEQUENCE DIAGRAMS.

FS2 : POINTER TO TABLE T TO BE ABLE TO LOCATE THE TASK OF THIS TASK, FS1(I), OF CURRENT FUNCTION

FS3 : PROBABILITY OF BRANCHING TO THE TASK POINTED BY FS4(I). PROBABILITIES ARE MULTIPLED BY 100 AND STORED AS INTEGER VALUES. PROBABILITIES MUST SUM UP TO 100.

FS4 : POINTER TO NEXT TASK IN THIS FUNCTION. ZERO POINTER ENDS THE DEFINITION OF A FUNCTION. NEXT TASK'S INDEX MAY BE HIGHER OR LOWER THAN CURRENT TASK INDEX.

Table 5.1 : Function Sequence and Function Pointer Tables Description (FS) and (FP)

TABLE (T) : TASK INFORMATION TABLE

EVERY COLUMN OF T IS DEFINED AS A SEPERATE VECTOR VARIABLE Ti REFERS TO COLUMN I.

T1 : TASK ID NO, XXDDDD
T2 : TASK STEP CODE, DDDD
T3 : I-BANK SIZE IN PAGES
T4 : D-BANK SIZE IN PAGES
T5 : FLAG FOR TASK AUTO-CREATION AT SYSTEM START-UP 0=NOT CREATED, 1=CREATED
T6 : NUMBER OF WAITING TERMINALS, AFTER WHICH ANOTHER COPY OF THE TASK WILL BE CREATED
T7 : MAX NUMBER OF COPIES ALLOWED
T8 : TASK KILL TIME 1 : IF AN ORIGINALLY CREATED TASK (EXCEPT AUTO-CREATED TASKS) DOES NOT EXECUTE WITHIN THIS TIME IT WILL BE KILLED
T9 : TASK KILL TIME 2 : AS ABOVE, BUT FOR TASK COPIES ONLY
T10 : NUMBER OF EXISTING TASK COPIES
T11 : STATE AND MEMORY ADDRESS OF EACH COPY OF THIS TASK GIVEN IN PAIRS, UPTO 3 COPIES STATE OF I'TH COPY OF TASK J IS IN T11(J,I*2-1) STATE VALUES ARE :
    0 : IN MAIN AND NOT BUSY
    1 : IN MAIN AND BUSY
    2 : SWAPPED OUT
    3 : BEING SWAPPED OUT
    4 : BEING SWAPPED IN
    5 : BEING CREATED
T11SZ : SIZE OF T11
T12 : NUMBER OF TRANSACTIONS WAITING ON THIS TASK
T13 : QUEUE TO STORE TRANSACTION ID NUMBERS OF UP TO 5 WAITING TRANSACTIONS I'TH TRANSACTION ID FOR TASK J IN T13(J,I)
T13SZ: SIZE OF T13
T14 : TASK MAX. EXECUTION TIME
T15 : NOT USED NOW
T16 : NOT USED NOW
T17 : NOT USED NOW
T18 : INFORMATION ABOUT FILES ACCSSESED BY A SPECIFIC STEPCODE. MAX. OF 10 FILES ACCSSESABLE BY ONE STEPCODE. 31 ENTRIES.
T20 : SAVE OF JID OF DIFFERENT SCTs.
T21 : SAVE OF JID OF THE FCT.

Table 5.2 : Task Information Table Description

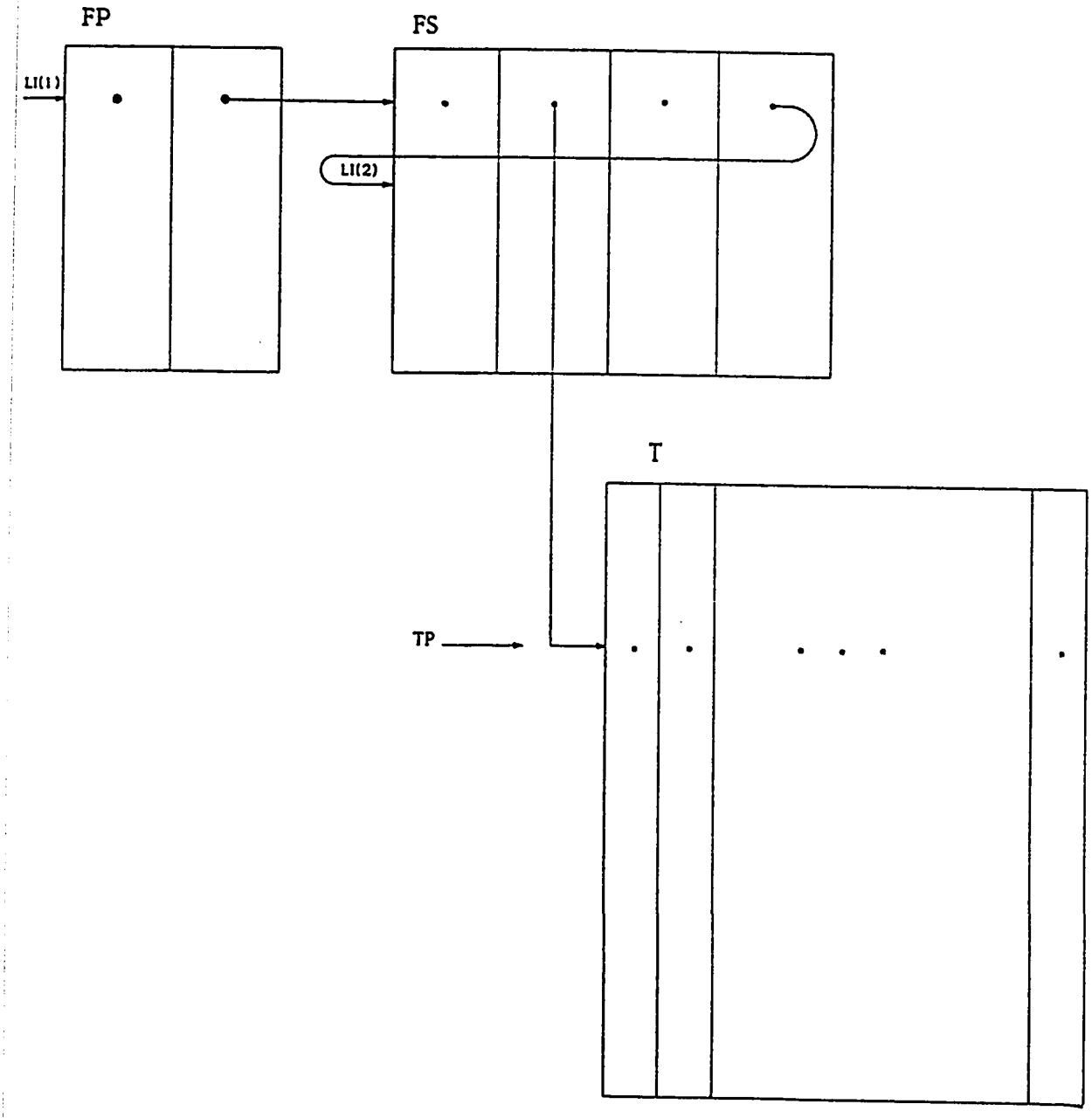


Figure 5.6 : Data Structures Relationship

5.3.2. Initialization Files

To incorporate the details of the SAMIS Online System software components in the model, and according to the methodology described for the abstraction of the file services methods of the operating system explained in section 5.1, three initialization files have been created including information about the details of the SAMIS Online System software components. These files were necessary inorder toenable the model to simulate the exact actions the SAMIS task encounters. These files are concerned with the knowledge about the sequencing of task invokation and database file accesses done by different tasks.

The first data file has been compiled to incorporate the possible sequencing of control transfere from one stepcode to the other in the execution procedure of the functions in the Civil Registration element of the SAMIS Online System. This data file includes the function identity number, followed by the first stepcode to be executed in this function when it is invoked, followed by the probability of transfer from this stepcode to other stepcodes, with a list of these possible stepcodes. If the probability was a zero, that indicates the termination of function execution. The detailed description of the file follows :

1. The first field is the stepcode of the first task to be executed for the designated function.

2. The second field contains one hundred times the probability of control passing to other stepcodes.

3. The third field contains the stepcode of the next task to which control of execution will be passed with the probability given in the second field.

4. A blank in the first field implies that there are more than one successor tasks to the task which appeared in previous first field.

5. A (-1) at column one, means the end of the description of the tasks for the current function.

6. A second (-1) at the end appears to indicate the end of functions in the current element.

Appendix (B) has more information about the contents of this file.

The second data file has been prepared to store information about files accessed by each SAMIS task. This information include which files are accessed, how many times the files are to be accessed, and by which task. This data file contains the first stepcode of the first function on a seperate line, followed by the possible files that are accessed by that stepcode (indented, not starting

on column one), followed by the expected number of file accesses. The file name list is continued, each file on a separate line, until no more files are accessed by the stepcode, where the new stepcode is just entered, starting on column one. This is continued until the end of stepcode-file description, for the current function is complete, where a special marker is then placed. This procedure could be repeated for the other elements. For more datails and a listing of this file refer to Appendix (D).

The third data file has been compiled to contain all relevant information about the MANAGE files used by the SAMIS Online System. Information such as file name, file type, expected file size, record size and more was included in the file. This file includes details of characteristics of files accessed during the execution of stepcodes in the SAMIS Online System. More information and a listing of the abstract file is listed in Appendix (A).

The information that is provided by these files is very frequently needed by the model during the simulation runs. When a function request is made and the transaction enters the model, it would be very inefficient to look for this information in the external files. Therefore, when the model is initialized it extracts the information from these input files and initializes the user node data structures before any processing is done.

To enhance the efficiency of the model a utility program has been developed and was documented. This program will access the file containing information about files accessed by stepcodes in functions of SAMIS Online System, and the file information properties file. From these two files a new file will be created joining both files, and incorporating some calculations to predict the average number of drum and disk accesses per task. The file that is produced by the utility program is used by the model to initialize internal data structures to be used during model execution. The file produced by the utility program will have the following structure.

First the stepcode is listed, followed by the average count of files this stepcode will be referencing. Information about all files accessed by this stepcode follow. If the file was indexed, the drum number on which the file index resides will be listed, then according to the mathematical formulae described in the abstraction of the SAMIS Online System software system components, the average number of drum accesses that are to be made to obtain the index from the index file residing on that drum is listed. The average number of times the file is to be accessed is listed to enable the model to simulate the multiple file accesses done by the stepcode.

The description of the second file is listed in the same format. The description of a maximum of ten files

that are accessed by any stepcode at one time could be accomodated by the model. A simple parameter change enables the model to account for more files.

If the file is unkeyed or hashed there would be no need for drum input and output operations, since it is assumed that any drum input or output operations are made only to obtain a key for an indexed file, thus there would be no entry indicating the drum number to be accessed in the file description field. A (-1) will be entered to indicate that only a disk input and output operation is to be done. There is no need to specify disk number since as discussed earlier, any disk input and output request will be randomly routed to one of the available disk drives.

5.3.3. Flow Control.

The model described in this section is an improvement of an old DISK/DRUM model that was built during the early stages of the work in this project. The old model was built with no regard to intertransaction communication, it simply simulates transactions coming in a system of Disk, Drums, memory units, and CPU's, treating them all similarly. This was not an accurate model, and results did not reflect any operational problems that the SAMIS system was actually facing. To remedy the problems encountered in the old model, it was clear that the relation between the actual transactions of the SAMIS system, how they are initiated, what tasks they invoke, how tasks operate in the SAMIS

system, and the PAWS transactions in the model need be identified and represented.

The model simulates SAMIS transactions from the moment they are initiated, to the moment they get terminated, if they ever do. The same operations, to some extent, that the SAMIS transactions encounters are applied to the PAWS transactions. When a user issues a request (invokes a function in any of the subsystems) the model reacts to this action as follows.

A function transaction is created and enters the system, this transaction is called the Function Control Transaction (FCT). The FCT represents a function and is responsible for control handling of the operations done by the function.

When the FCT enters the system, simulating a real SAMIS transaction, it will be routed to the user node, and the model will look for the first task the function is expected to do, then it checks the state of the corresponding transaction for that task. Each transactions will be associated with a driving task called the Step Code Task (SCT).

The SCT will be associated with the task and will execute all the requirements of that task. If there is no SCT task available, the model will try to create another copy (see restrictions in Appendix C). If no copy

could be created, the FCT will try to swap in any swaped out SCT copy, if that also does not work, then the FCT will be put on hold, until an SCT task copy becomes available.

If it is possible to create another copy of the SCT task, then the model will create an SCT task copy and start simulating the actual cost of task creation done by the SAMIS system, involving some CPU, disk and drum input and output operations. Memory will be allocated to the SCT copy depending on the nature of the task the SCT copy is supposed to be associated with.

If the SCT original task and all of the other copies were busy, (see Appendix (E) for the timing diagram of transactions in the PAWS model) the FCT waits on a queue until it is notified (interrupted) that the SCT task copy is free to service it. It then gives control to the SCT task copy, and goes into an FCT pool and waits until the SCT task copy finishes the processing requirements of the task and notifies it. The FCT will then take over and continues on the requirements of the function until no more requirements exist indicating the end of function processing where the FCT will be released .

When the SCT task copy gets the control, it is scheduled for CPU execution. When its turn comes, it executes for some time, depending on the expected

execution time, and the CPU time quantum allocated . If its execution time is more than the quantum, it will be taken out of the CPU, and rescheduled later.

This cycle will be repeated until the SCT finishes CPU execution, or issues an I/O request. If no I/O is requested, the SCT task copy gives the control back to the FCT, and goes to wait in an SCT pool, where it becomes available for use by any other requesting SAMIS task.

If there are any I/O requests by this transaction, then the I/O cycle will be initiated, and depending on the specific SAMIS task, Disks and Drums are accessed.

In the SCT pool, SCT task copies wait until a task requests the association with one of these SCT task copies, then the FCT will pass the control to the requested SCT task copy. If no such tasks arrive after some preset time (a very rare situation) the SCT will be forced out of the SCT pool, requested to release all of the memory allocated to it so that other SCT task copies could be created, and are able to use the memory.

There are two time periods involved here, the time given to the original SCT task copy to wait before being asked out of the SCT pool, which is called 'Kill Time 1', and the time given to all other SCT task copies, called 'Kill Time 2'. More information could be found in Appendix (C).

The transaction in a PAWS model is routed in the

IPG exactly as if it were a SAMIS transaction being routed in the actual system. There is a very close resemblance between the FCT,SCT and tasks in the SAMIS environment [25].

5.4. Detailed IPG and PAWS Program

At this point the internals of the user node and the transaction flow control should be clear. The details of the special purpose section in the basic IPG could now be described. The details of the special purpose component of the basic IPG are shown in Fig. 5.7.

Transactions of the model are created according to the distribution and depending on the arrival rate of the actual SAMIS transactions. Different interarrival rates are to be tested to simulate different workloads on the system and study the effect on the performance.

Every created transaction will have its own identity, and will be associated with one of the SAMIS Online System functions representing a transaction arriving to the SAMIS Online System.

The generated transactions are the FCT transactions described earlier. They will be routed to the user node. When an FCT transaction enters the user node it is initialized, then according to the specific conditions and status of the system, and the requirements of the individual

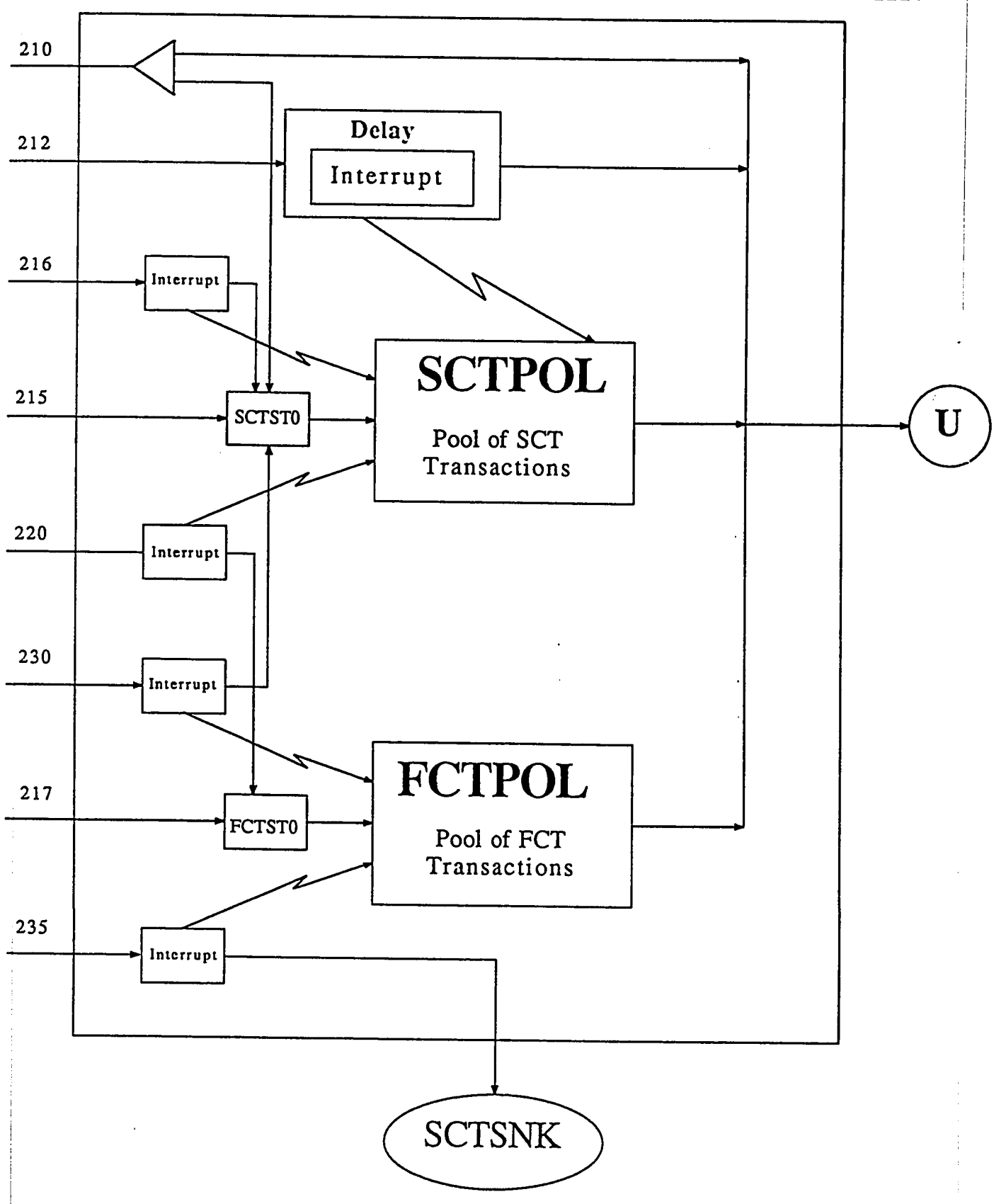


Figure 5.7 : Special Processing component Details

transaction, different transactions (namely SCT transactions) representing the incoming SAMIS Online System transaction are initiated from the user node to be routed through the IPG according to a "phase number" set by the user node. Phase numbers identify the next operation the FCT transaction is to carry out in the model.

Every possible place the task could go to from the user node, is marked with a special number. According to its phase number the SCT transaction will be routed to a specific location to perform some operation, such as disk I/O, drum I/O, memory allocation, CPU cycle, etc. A return address is saved for each task getting out of the user node, so that it is possible for the task to know what to do next, when it returns to the user node.

There will be many tasks active in the system at a given time, therefore there will be many transactions returning to re-enter the user node asynchronously. Transactions getting out of the user node could go to do disk I/O, drum I/O, CPU cycle, acquire memory, release memory, wait for its turn for CPU, wait for a certain condition to occure, or get sunk.

The user node is the module that is responsible for routing PAWS transactions in the model. The operation of the user node simply, begins with an initialization phase that each transaction is going to face, only once. Then, it is passed through a task creation "MACRO", again done

only once in the lifetime of a specific transaction. The rest of the user node is organised in a structured manner, as a set of MACROs. Each Macro is devided into steps, and every step is responsible for doing a specific job.

When a transaction re-enters the user node subsequently it carries a return address. The specific step number in one of the Macros is identified from this return address, and the transaction is routed to that step.

In the special processing section there are two main queues. The queues are actually associated with servers as a SERVICE node in PAWS, but the servers are not of any importance to our purposes. The requirement is for a representation of the operation of waiting until a certain condition occurs. SERVICE nodes were used to implement these queues, and the power of these service node is set to zero. Thus transactions will wait in these queues forever unless they are abnormaly forced out of the queues, which is the action to be done as will be discussed shortly.

The two queues actually represent two pools of transactions. The first is the FCT pool where all FCT transactions wait, and the second is the SCT pool where all the SCT transactions wait.

Whenever the FCT transaction gives control to the SCT transaction, the FCT will be directed to the FCT pool until the SCT transaction finishes up its processing and decides

to give control back to the FCT. At that time, the SCT will wake up the FCT transaction. This will be done when the user node routes the SCT transaction with a phase number of 230 or 235, and supplies the PAWS transaction identity number (JID) of the FCT to be waken up. This JID was saved in one of the data structures before the FCT was directed to the pool.

Passing with phase number 230 or 235, the SCT will initiate an interrupt addressing the specific FCT that is waiting in the FCT pool, and forcing it to go out of the pool to the user node. The FCT saves a return address in its structure that would enable it to know the route it should take when returning to the user node.

Similarly if control is to be passed from any FCT transaction to any of the SCT transaction waiting in the SCT pool, the user node should pass the SCT transaction PAWS identity number (JID) that was saved in one of the data structures before the SCT was sent to the pool. The FCT will then be assigned a phase number of 216 or 220 depending on the situation, then the FCT will be sent to the special processing section, where it causes the SCT to be aborted from the SCT pool and routed to a pre-known return address in the user node.

These pools are used to switch control between the SCT transactions and the FCT transactions. Whenever the user node decides to get rid of any of the the SCT

or the FCT transactions, it would direct them to the IPG with phase 200 or 221 respectively. A summary of phase numbers and the associated meaning is listed in Table 5.3.

This would conclude the discussion about the construction of the detailed IPG and the design of the PAWS program to produce the required PAWS model of the SAMIS Online System. For an overview of the complete IPG of the SAMIS Online System model refer to Fig. 5.8. Fig. 5.9 shows all the files used by the model and Fig. 5.10 shows all files produced by the model.

```
100 : Produce Random Numbers
200 : Sink SCT Transaction
201 : Allocate Memory
202 : Release Memory
203 : CPU Queue and Service Node
204 : Terminal I/O Handling
205 : Drum I/O Processing
206 : Disk I/O Processing
209 : CPU Scheduler
210 : Place SCT on SCTPOL & Return to User Node
212 : Delay
215 : Place SCT on SCTPOL
216 : Interrupt SCT From SCTPOL & Join SCTPOL
217 : Place FCT in FCTPOL
220 : Interrupt SCT & Join FCTPOL
221 : Sink FCT Transaction
```

Table 5.3 Phase Numbers

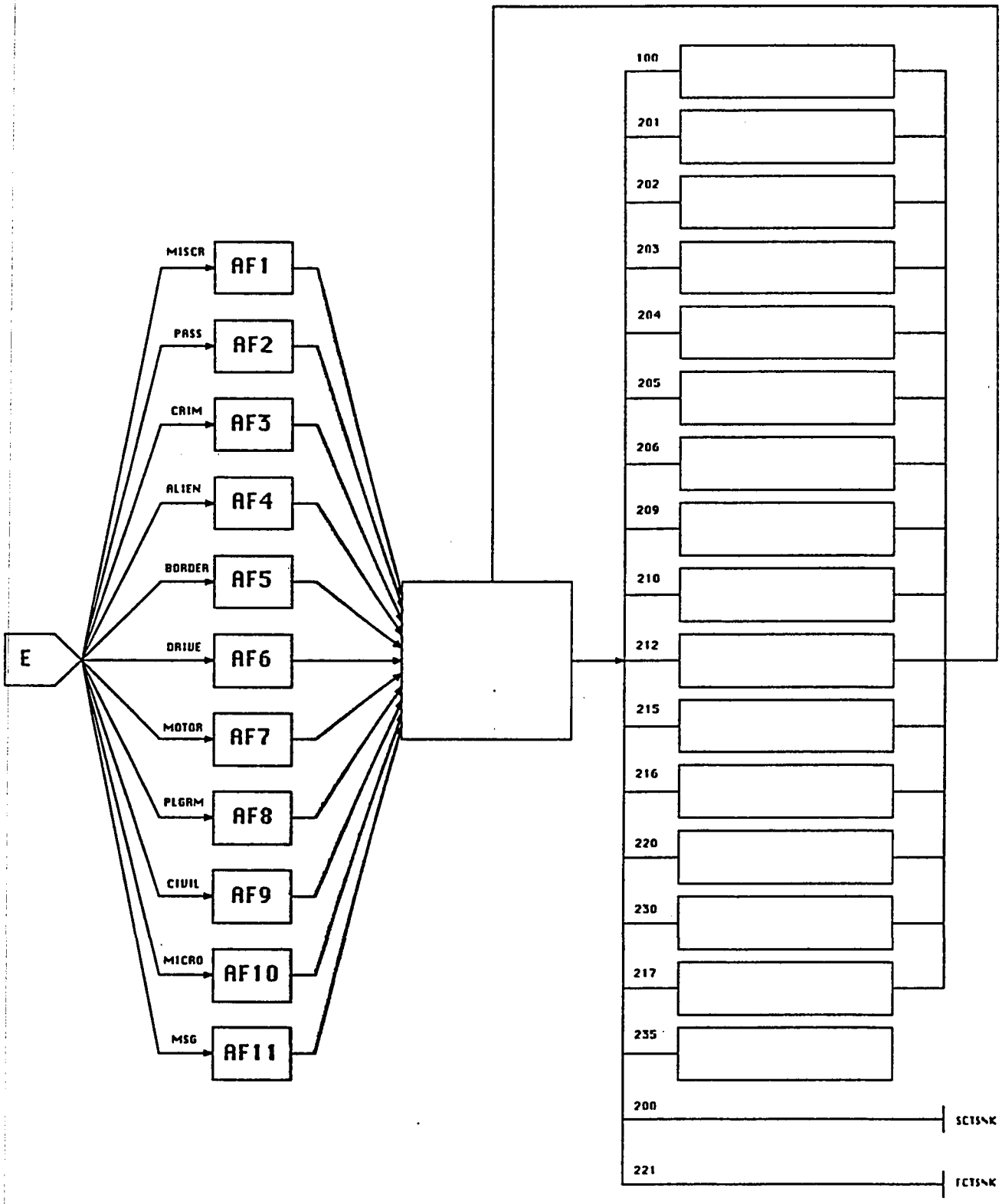


Figure 5.8 : Overall View of the IPG
SAMIS Online System Model

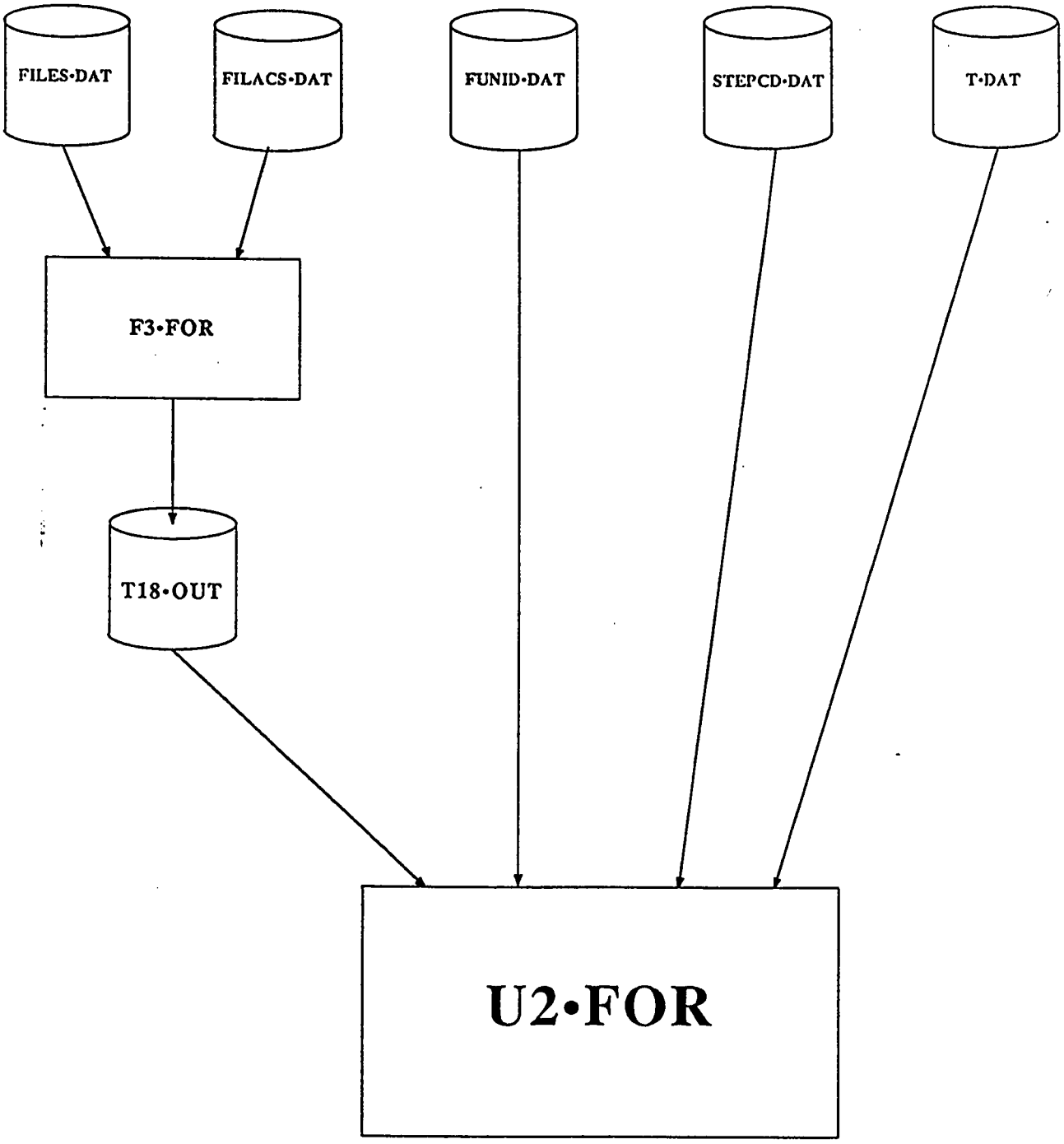


Figure 5.9 : Files Used by the
SAMIS Online System Model

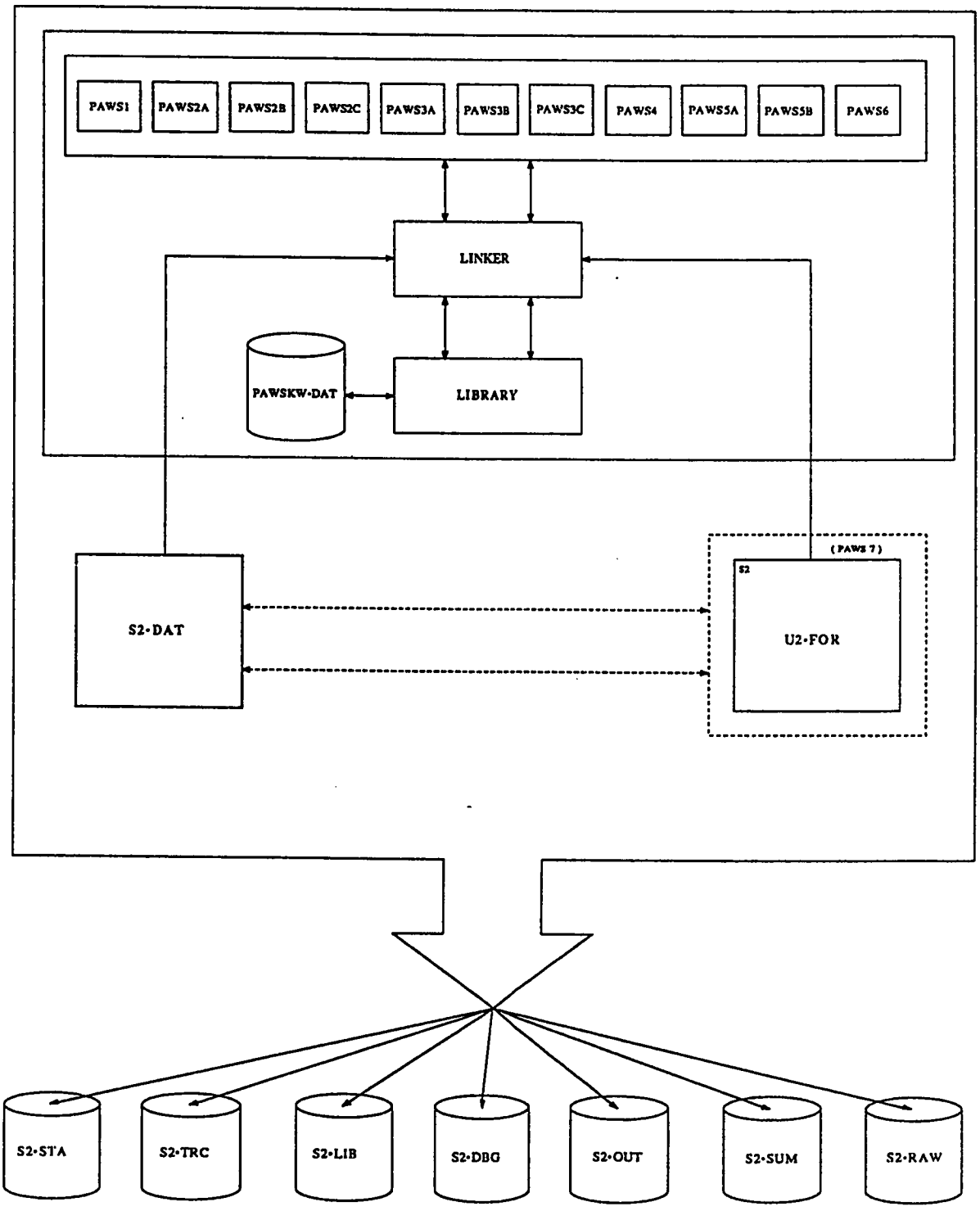


Figure 5.10 : Files Produced by the SAMIS Online System Model

6. EXPERIMENTATION, ANALYSIS AND RESULTS

6.1. Experimentation

The SAMIS Online System PAWS model was built after a careful study of the basic parameters which are expected to affect the system performance. These parameters include main memory size, CPU quantum time, and input transaction load which enabled the testing of the model under different conditions. A factor that might also affect the correctness of the results is the simulation duration. The effect of simulation time on the behaviour of the model has also been studied.

The experimentation consists of varying model parameters within certain ranges and observing the behaviour of the model. Satistics have been collected and a number of graphs showing different performance aspects have been plotted. Statistics include CPU, disk, drum, channel and memory utilizations, mean queue length for CPU, disk, drum, and memory, in addition to the mean response time showing the average time spent by a SAMIS transaction in the system.

A total of five experiments have been conducted and these experiments are described in the rest of this section.

The first set of experiments was concerned with the study of the effect of changing CPU quantum time and the transaction flow rate on the system, while keeping memory

size fixed at 2282 pages and simulation time of 120,000 time units. One simulation time unit actually represents one millisecond of real time.

Different CPU quantum settings included 25, 50, 100 and 200 milliseconds. CPU time required by a transaction is assumed to be uniformly distributed in the closed range [0,200] milliseconds. Different transaction flow rate settings that are used are 4, 8, 16, 24 and 32 transactions per second. Flow rate has been represented as transaction interarrival time in the model, resulting in interarrival times of 500, 250, 125, 62.5, 41.67 and 31.25 milliseconds. It should be noted here that the actual observed transaction rate in 1983 was about three transactions per second without the Pilgrim Subsystem operating, and six transactions per second with the Pilgrim Subsystem operating. In the second year, the rate increased to five transactions per second without the Pilgrim Subsystem, and eight transactions per second with the Pilgrim Subsystem [32]. Thus, the experiments conducted test the system under very high transactions load, exceeding the expected load fo 1990.

Statistics collected include CPU queue length, drum queue length disk queue length, mean memory queue length, CPU queueing time, drum queueing time, disk queueing time, CPU utilization, drum utilization, disk utilization, memory utilization, CPU mean service time, drum mean service time, disk mean service time, mean response time, in addition to

the utilization of seven channels. For every single combination of transaction flow and CPU quantum it took about 20 to 30 minutes of VAX-11/780 run time to carry out an experiment. Thus to conduct this experiment and collect relevant statistics, it took about 600 to 750 minutes of actual computer run time.

The second experiment was conducted for testing the effect of short simulation time and fixed memory requirements for different transactions. Simulation time was fixed for this experiment to 60,000 simulation time units, memory size was fixed to 2282 pages, I-Bank memory requirement was fixed at 6 pages, and I-Bank memory requirement was fixed at 4 pages per task copy. The CPU quantum was varied from 25, 50, 100 to 200 milliseconds, and the transaction flow rate was varied from 2, 4, 8, 16, 24 to 32 transactions per second. Actual run time spent to collect these statistics was about 600 minutes.

The third experiment was conducted to study the effect of simulation time on the model's behaviour. It is necessary to determine the time the model reaches a steady state. Thus the experiment was conducted by fixing the transaction rate to 32 transactions per second, where transactions came with interarrival rate of 31.25 milliseconds. The memory size was fixed at 2282 pages and the CPU time quantum was fixed at 100 milliseconds. Simulation time was varied from '30,000', '60,000', '120,000', and '240,000' simulation

time units, where one simulation time unit is considered equivelant to one real millisecond. Detailed CPU, drum, disk, memory and channel statistics have been collected by this experiment.

The fourth experiment testing different memory sizes and the effect on the model's behaviour was conducted. Memory size was varied from 2282, 3423, to 4564 pages while fixing transaction flow rate at 32 transactions per second, CPU quantum at 100 milliseconds, and simulation time to 120,000 simulation time units, each time unit equivelant to one actual millisecond.

The fifth experiment was conducted to study the load on disk drives and drums. This experiment was conducted by fixing the transaction flow rate to 16 transactions per second, implying and interarrival time of 62.50 milliseconds between transactions, CPU time quantum has been fixed to 50 milliseconds, and the simulation time was 120,000 time units. During this experiment there were 1045 drum accesses, and 642 disk accesses.

A final note about experimentation should be mentioned regarding the sizes of internal buffers in the model, and in the PAWS language. It is not unlikely for the model to abort at a certain stage during the simulation process, for the reason of having one or more of the internal buffers full. This issue occures because it is not a trivial task to anticipate the sizes of these buffers before the model's

behaviour is studied and the model is run. If the model aborts at a certain time during the simulation process, it issues a certain message indicating the problem, and the buffer that is full, and recommending solutions. The problem should then be remidied, and it becomes necessary to recompile and link the complete package again, because these parameters define dimensions of buffers, and FORTRAN does not allow automatic dimensioning. This is a procedure that takes about 30 to 40 minutes of compilation and link times on a VAX-11/780 computer system.

6.2. Analysis and Results.

The study of performance modeling and analysis passes through different phases to be conducted in sequence. Statistics were collected after testing the model on one of the subsystems. The analysis phase in this project would be very lightly conducted due to the fact that it is not the basic intent of the work, and due to the need for incorporating more detail in the model's database to include the description of the other subsystems.

Thus during the analysis presented here, care should be taken not to reflect the bahaviour of the SAMIS Online System model, to the bahaviour of the complete SAMIS Online System application system, and noting that the statistics actually reflect the bahaviour of the model while running one of the subsystems.

Studying the queue formation on servers requires the study of statistics on queue lengths and queueing times. Fig. 6.1, 6.2 and 6.3 show the trend in queue lengths formed by transactions on the CPU, drum, and disk servers respectively. Fig. 6.4 shows the trend in queue lengths formed by transactions waiting on memory to be allocated.

Queueing time on servers also gives an indication of the service provided, and Fig. 6.5, 6.6 and 6.7 show the trend in queueing time on CPU, drums and disks respectively.

It is noteworthy here that the transactions flowing in the model represent the Civil Registration Application subsystem transactions flowing in the SAMIS Online System. A transaction invoked in any of the SAMIS Online System Application system is abstracted in the model as a sequence of CPU utilization, disk and drum input and output operations, and some memory access operations.

Having the detailed information about one of the application systems does not reflect actual behaviour of the model towards different transactions from different application subsystems. The only difference however between one subsystem and the other is the different files accessed by these subsystems, although different subsystems might access the same files. Actual operations and procedures executed by the operating system to handle requests from different subsystems are basicaly the same. That is why the analysis of the utilization of different servers are valid

to some extent, while the obtained utilization of drums for example does not reflect the behaviour of the system when under demand from all the application subsystems.

It could be seen from Fig. 6.37 showing the average number of accesses per drum, that some drums are highly utilized, about 400 accesses during a simulation time of 120,000 milliseconds, while other drums are not utilized at all during that period. Fig. 6.38 shows that disks on the other hand are very uniformly utilized, and that no problems are observed in their utilization. It could be concluded from this observation that drums are causing performance degradation of the overall system, by having large queue formations waiting for the service of a specific drum, while other drums are rarely utilized. It should be noted that this is due to the intense access of some files that cannot be accessed without the use of specific drums. Performance graphs representing the CPU, drum, disk, memory along with the utilization of all channels are shown in Fig. 6.8 to Fig. 6.18.

Fig. 6.19, 6.20 and 6.21 show the mean service times offered by different servers, namely the CPU mean service time, drum mean service time, and disk mean service time respectively. Fig. 6.19 has an obvious and clear meaning where the CPU service time is very close to the CPU time quantum. Fig. 6.20 shows that drum service time goes to less than about 8 milliseconds at high transaction rate, and

figure 6.21 shows that average disk service times tend to reach about 40 milliseconds at high transaction load.

The mean response time shown in milliseconds in figure 6.22 shows that for high transaction load, the average response time oscilates between 2 to 3 seconds. Communication subsystem delays are not incorporated in detail in the model structure.

If a model is representative of the modeled system, then it should be able to simulate the system in continuouse operation. Experimenting with the model for very short periods of time might not produce credible results. The time period elapsing at the initiation of opeartion of any system, called the transient period, does not reflect actual system behaviour. It represents the period where startup delays and initialization procedures that are not usually encountered during normal system operation are done.

The statistics obtained from the model should show the occurance of the transition from the transient state to the normal operation state, called the steady state of the system. Performance statistics should be collected after the system is in a steady state, thus experiments should be conducted to determine the time the model reaches the steady state. These experiments are conducted by fixing all parameters and varying the simulation time, and studying the trend in the model's behaviour. At that time it is possible to detect the transition time from transient to steady

states.

Experiments were conducted and statistics were collected by fixing the transaction flow rate at 32 transactions per second, CPU quantum to 100 milliseconds, and memory size to 2282 pages.

The effect of simulation time on the behaviour of different servers and other elements of the model is shown in Fig. 6.23 to Fig. 6.36. These graphs show that statistics collected at times before 80,000 time units (milliseconds) does not reflect actual model's behaviour at normal operation. The steady state of the model is reached at about 100,000 milliseconds, and statistics collected after that time are representative of the system's behaviour under normal opeartion conditions.

Thus running the model for 100,000 time units, or 500,000 time units should not produce different perfromance results. This would explain the reason for the unexpected perfromance measures encountered in the second experiment where simulation time was set to 60,000 time units.

Finally to study the effect of the size of the main memory on the performance of the model, the statistics collected by the fourth experiment have been plotted and appear on Fig. 6.39 to Fig. 6.43. These graphs show groups of plots that describe different performance measures. Fig. 6.39 shows the plots describing the effect of memory size on

the queueing time of the CPU, drum, and disk. Fig. 6.40 shows the effect of memory size on the length of the queue formed on the CPU, drum, disk and main memory. Fig. 6.41 shows the effect of main memory size on the utilization of the CPU, drum, disk and main memory. Fig. 6.42 shows the effect of main memory size on the mean service time of the CPU, drum and disk, along with the mean response time for an average transaction in the system. Finally Fig. 6.43 shows the effect of main memory size on the utilization of different channels.

From these graphs, it is observed that the larger the memory gets in size, the better the utilization of different components. This is observed through Fig. 6.41 and Fig. 6.43. From the plot of the mean response time in Fig. 6.42, it could be observed that the larger the memory size, the better the response time, and thus the better the performance. From Fig. 6.40 showing the different queue length on different components it could be concluded that the bigger the memory, the shorter the queue length, thus the better the response time. Fig. 6.39 shows that the bigger the memory the shorter the queueing time on different components.

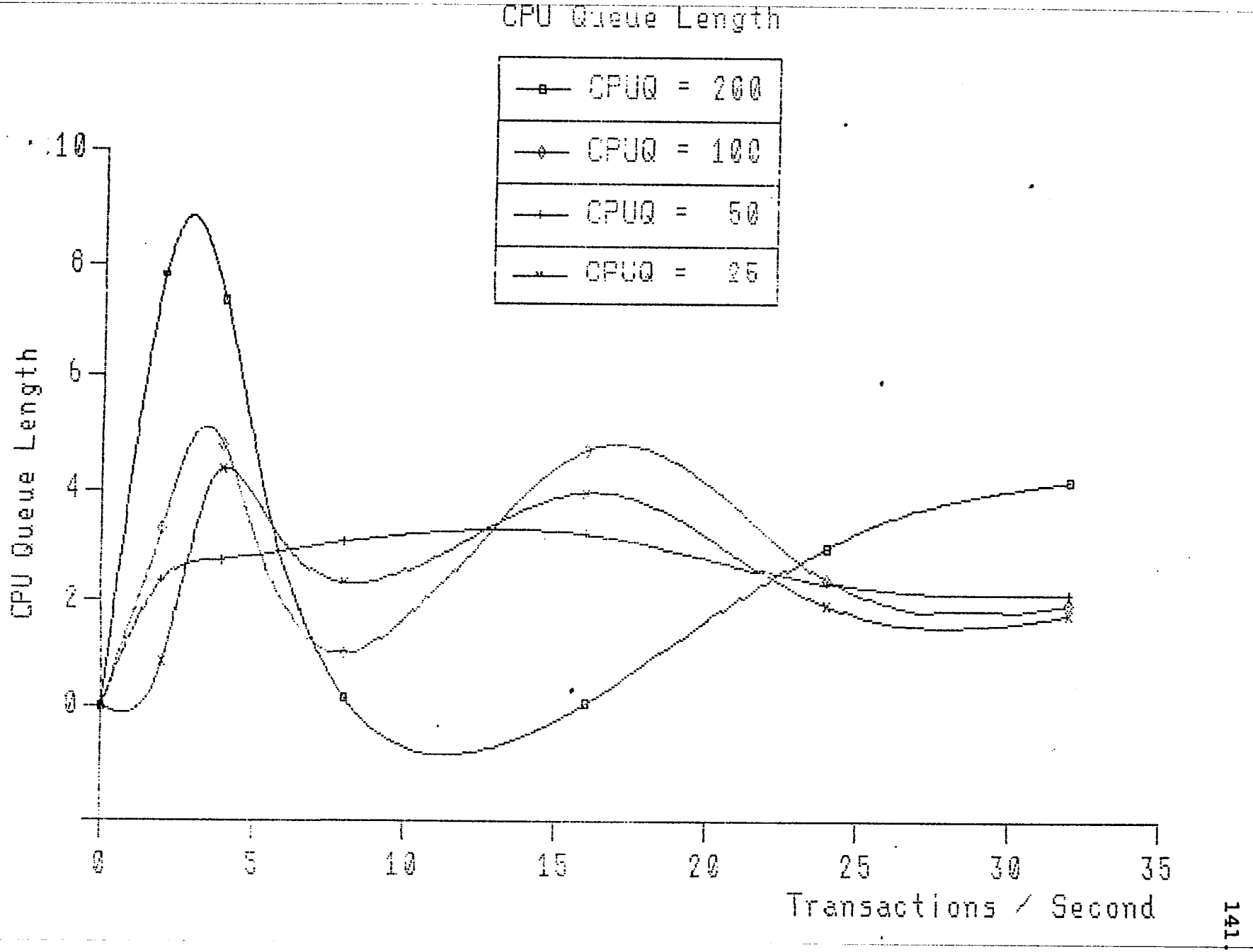


Figure 6.1 CPU Queue Length

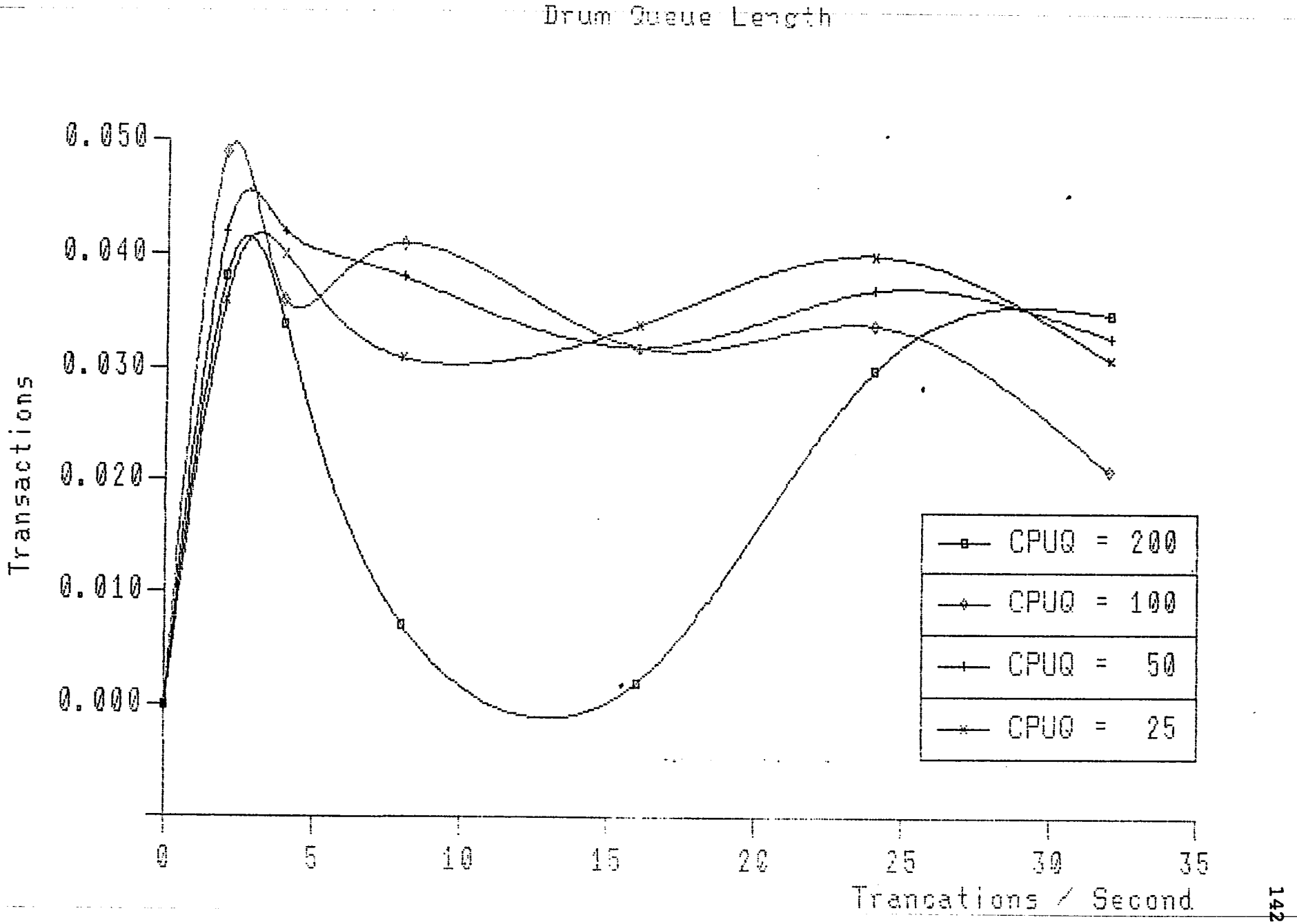


Figure 6.2 Drum Queue Length

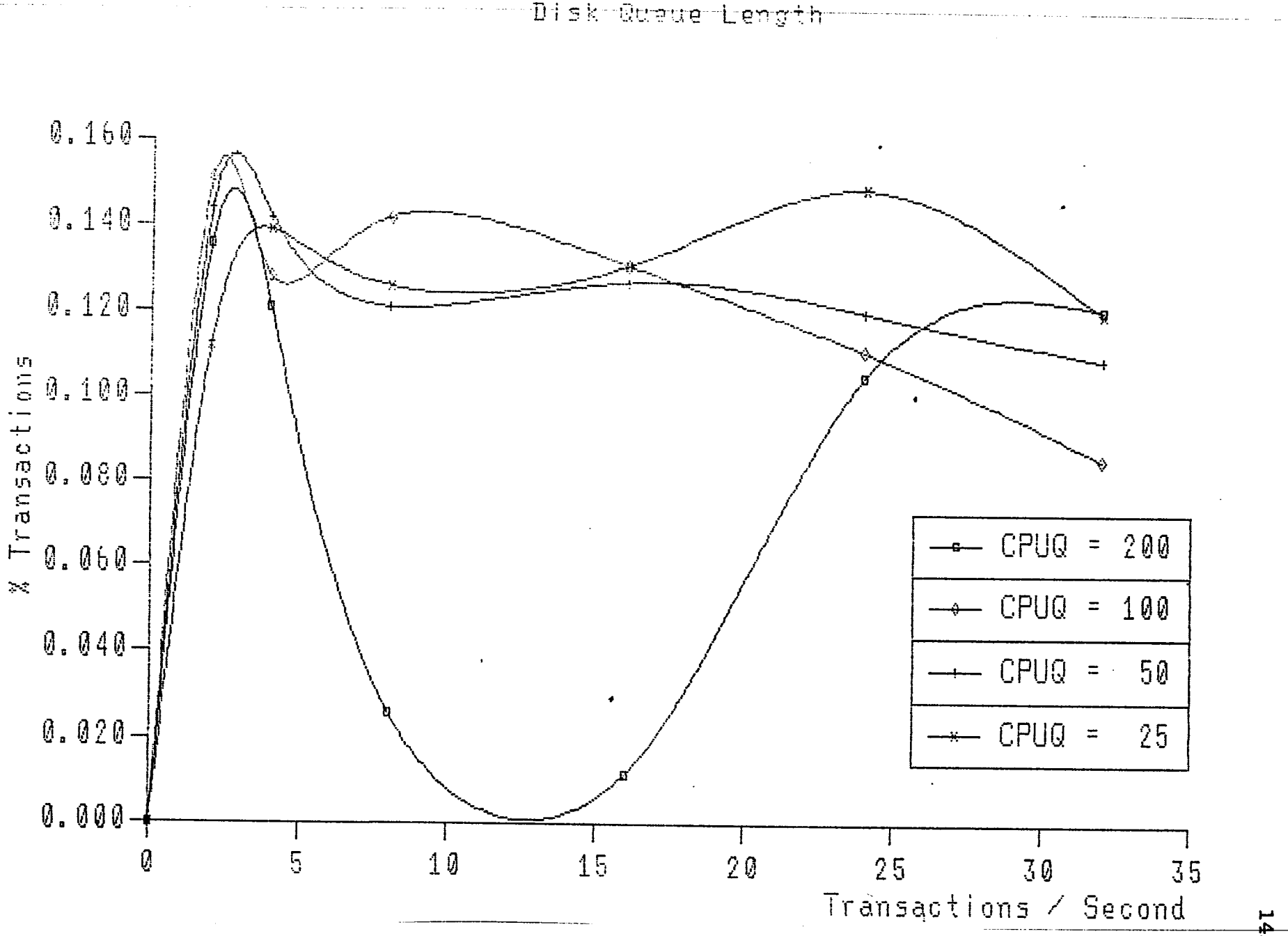


Figure 6.3 Disk Queue Length

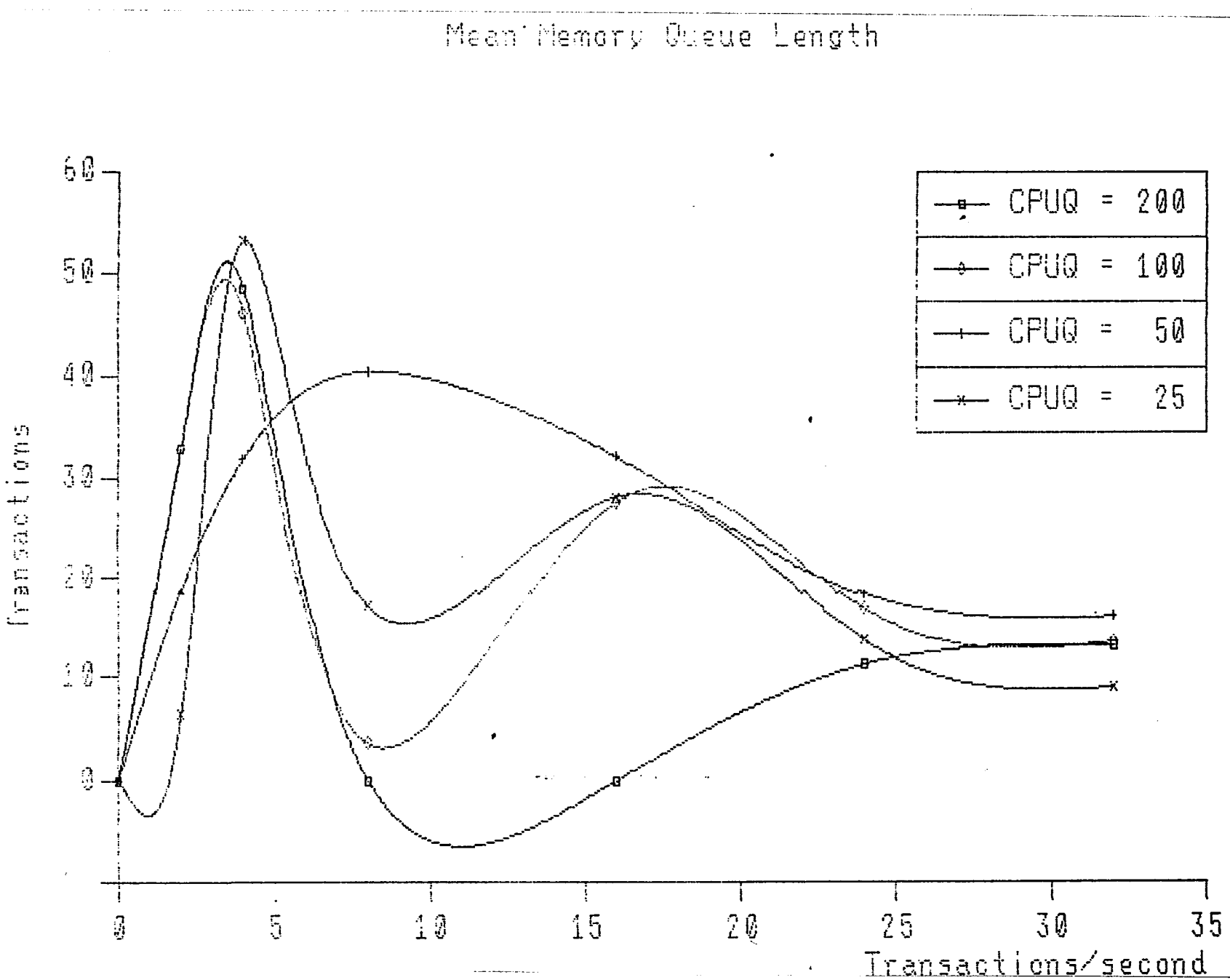


Figure 6.4 Mean Memory Queue Length

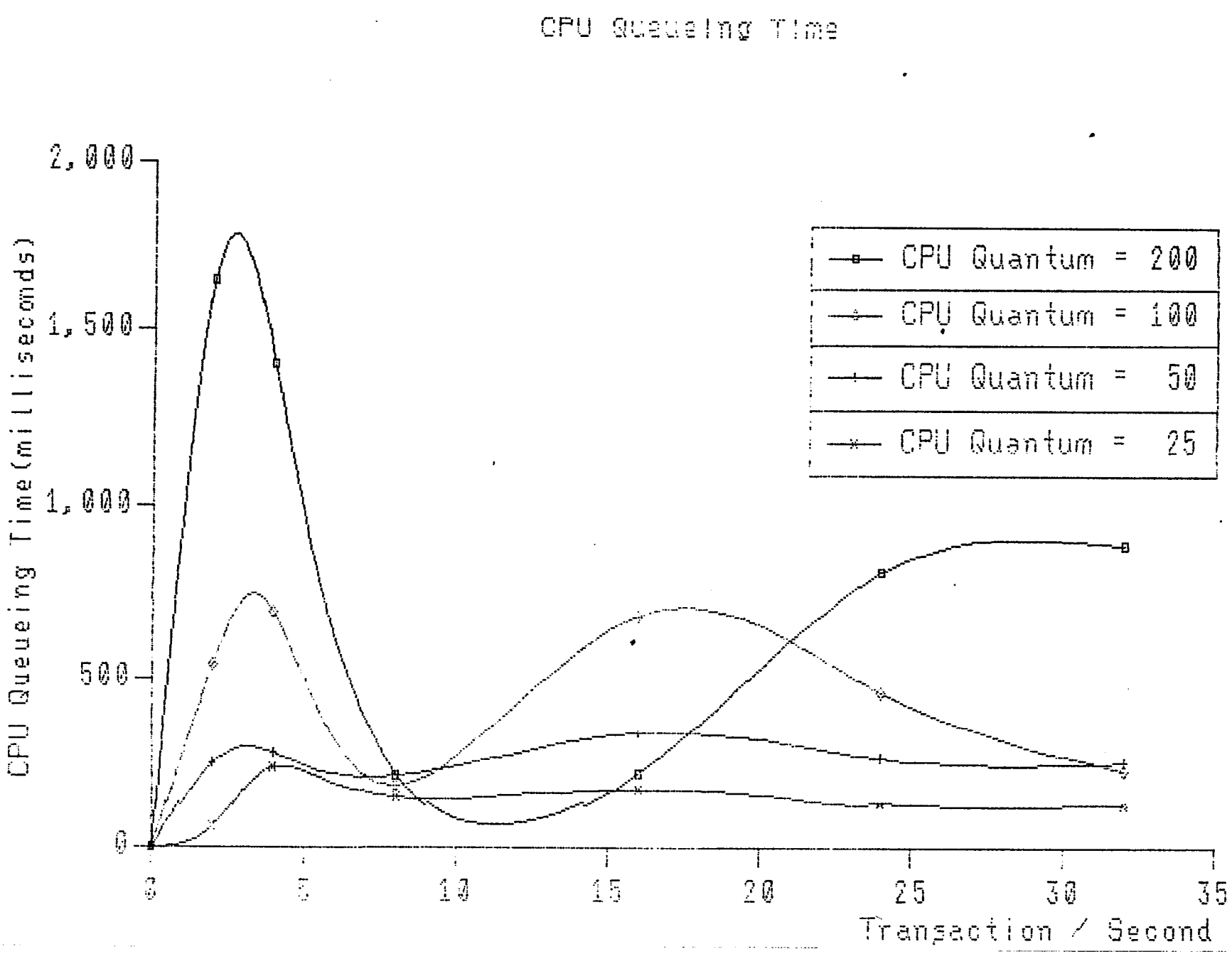


Figure 6.5 CPU Queueing Time

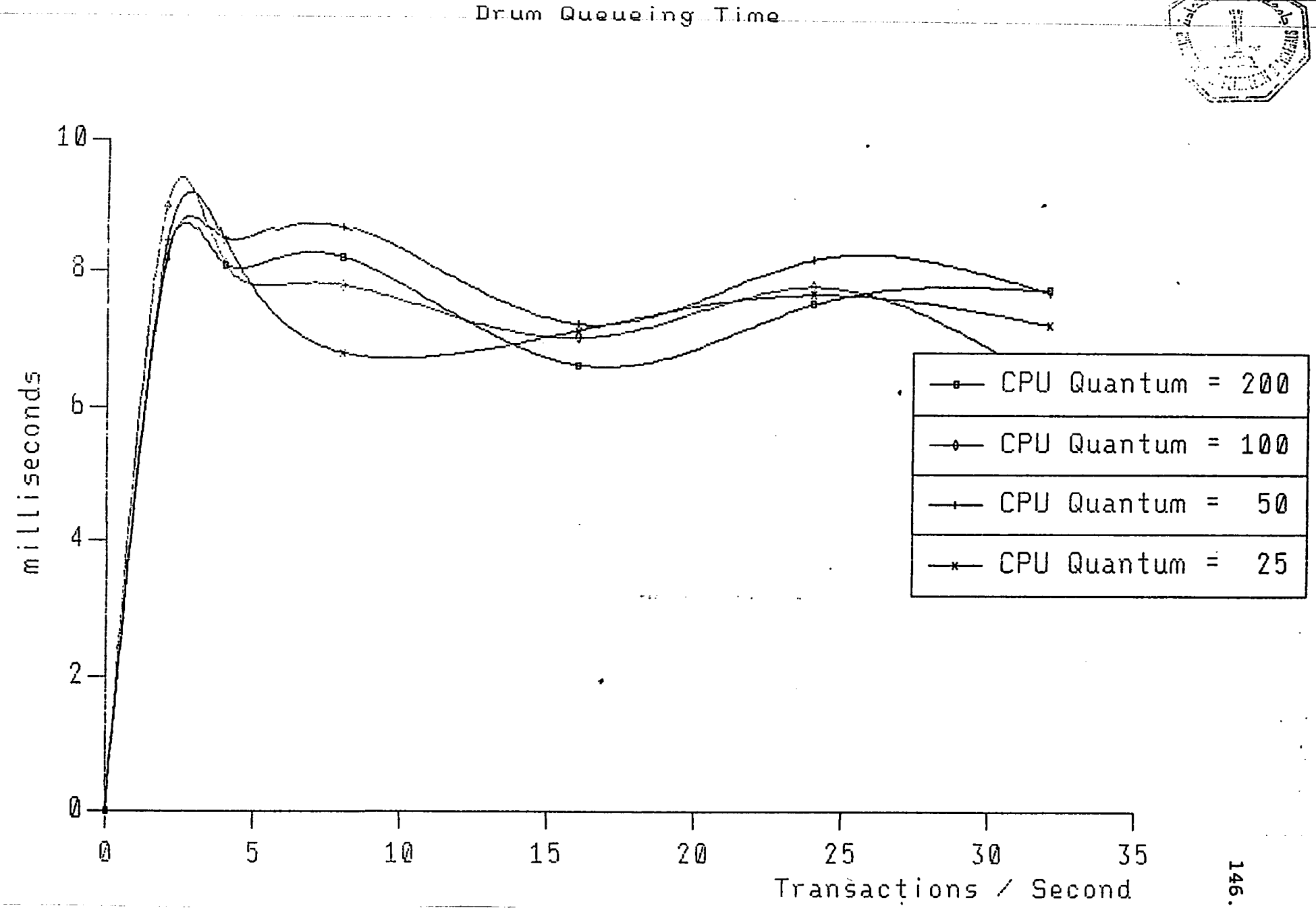


Figure 6.6 Drum Queueing Time

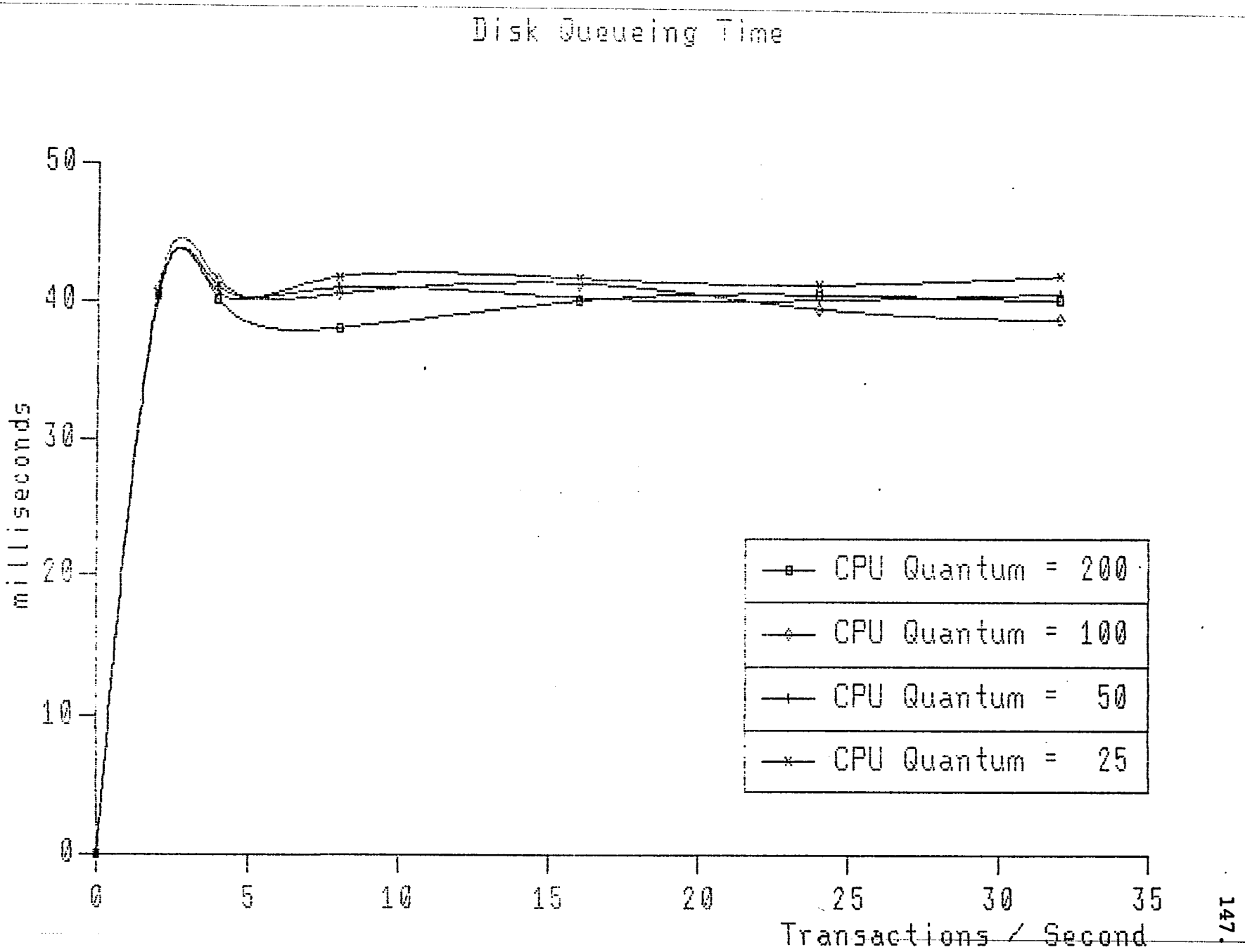


Figure 6.7 Disk Queueing Time

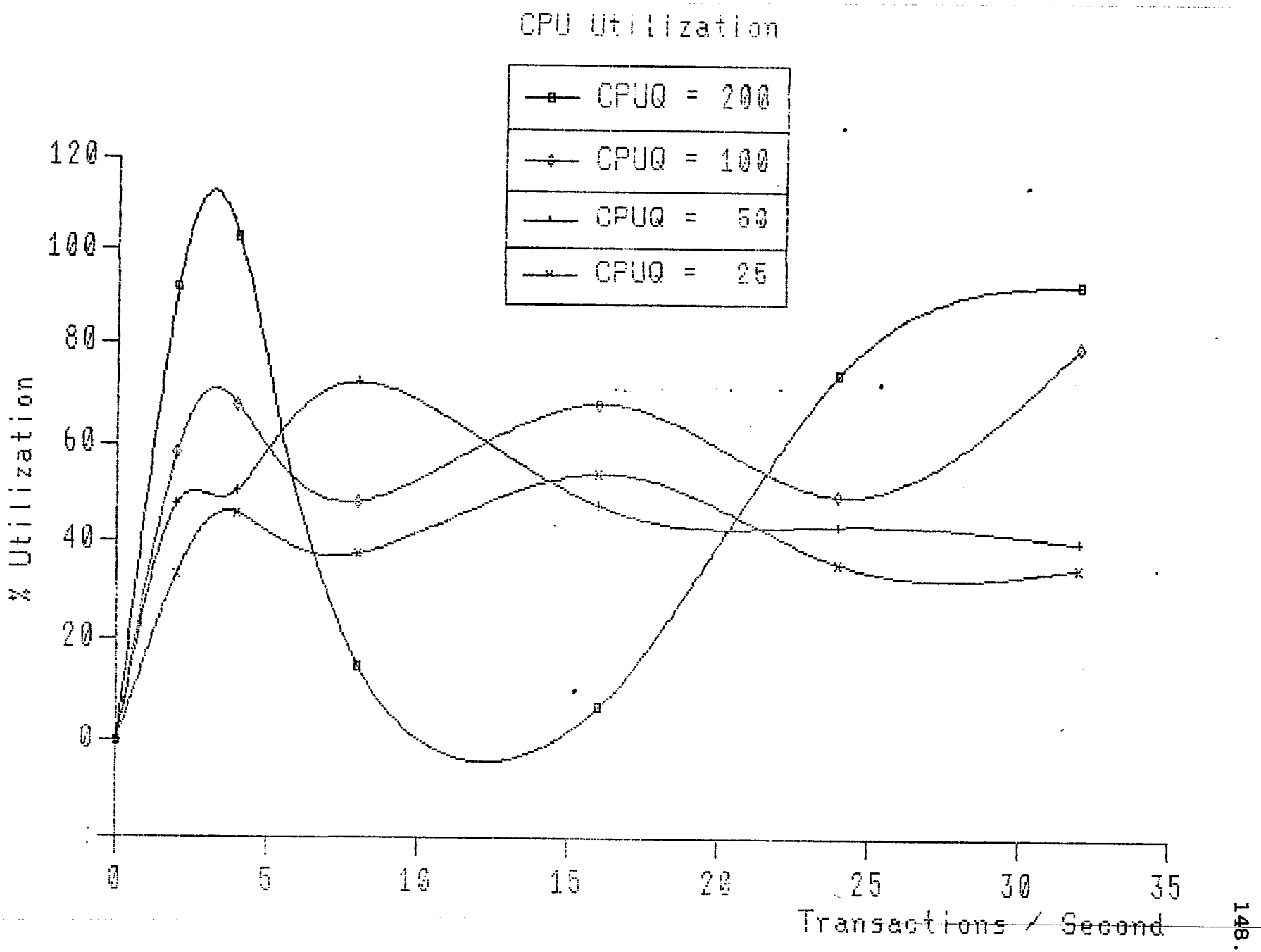


Figure 6.8 CPU Utilization

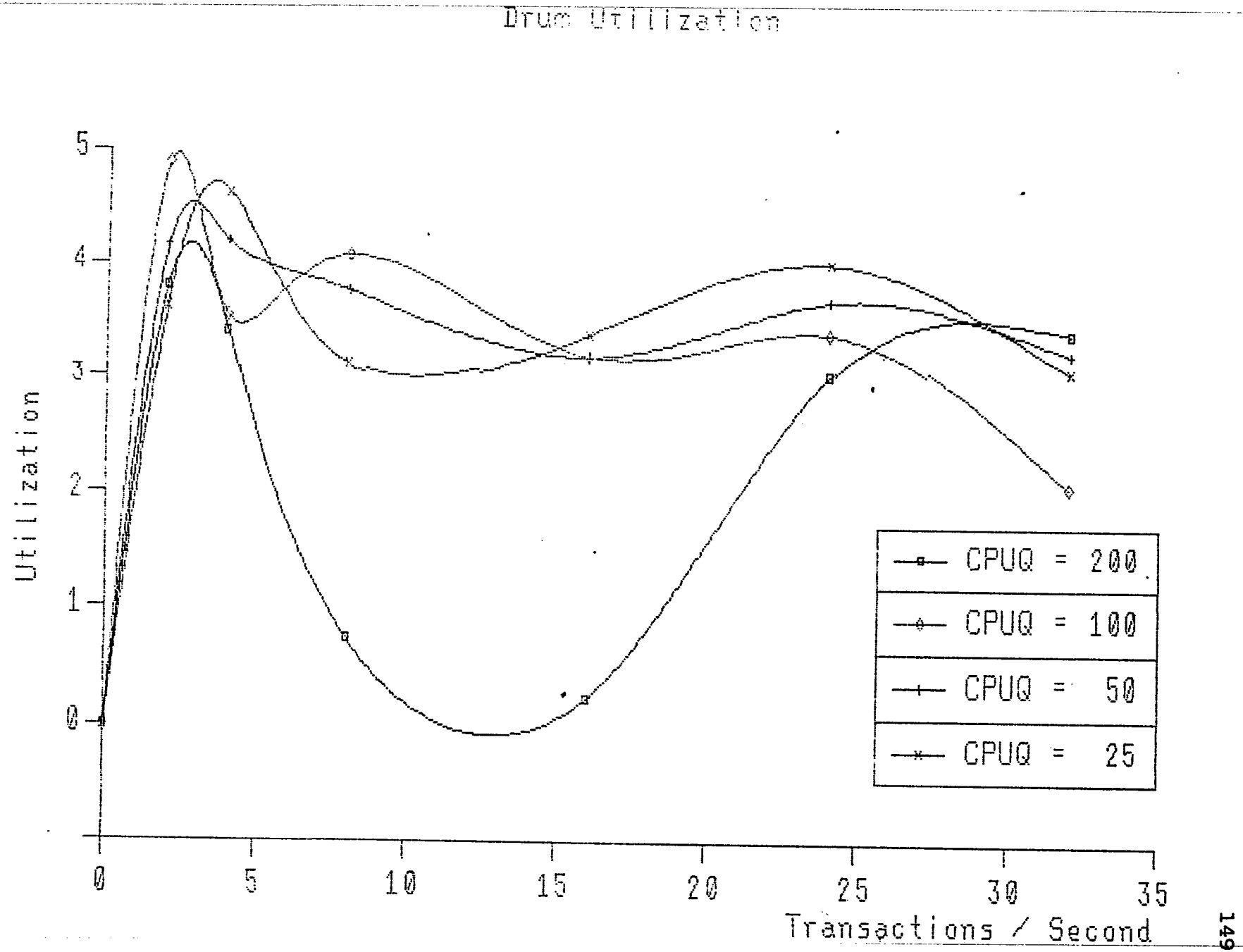


Figure 6.9 Drum Utilization

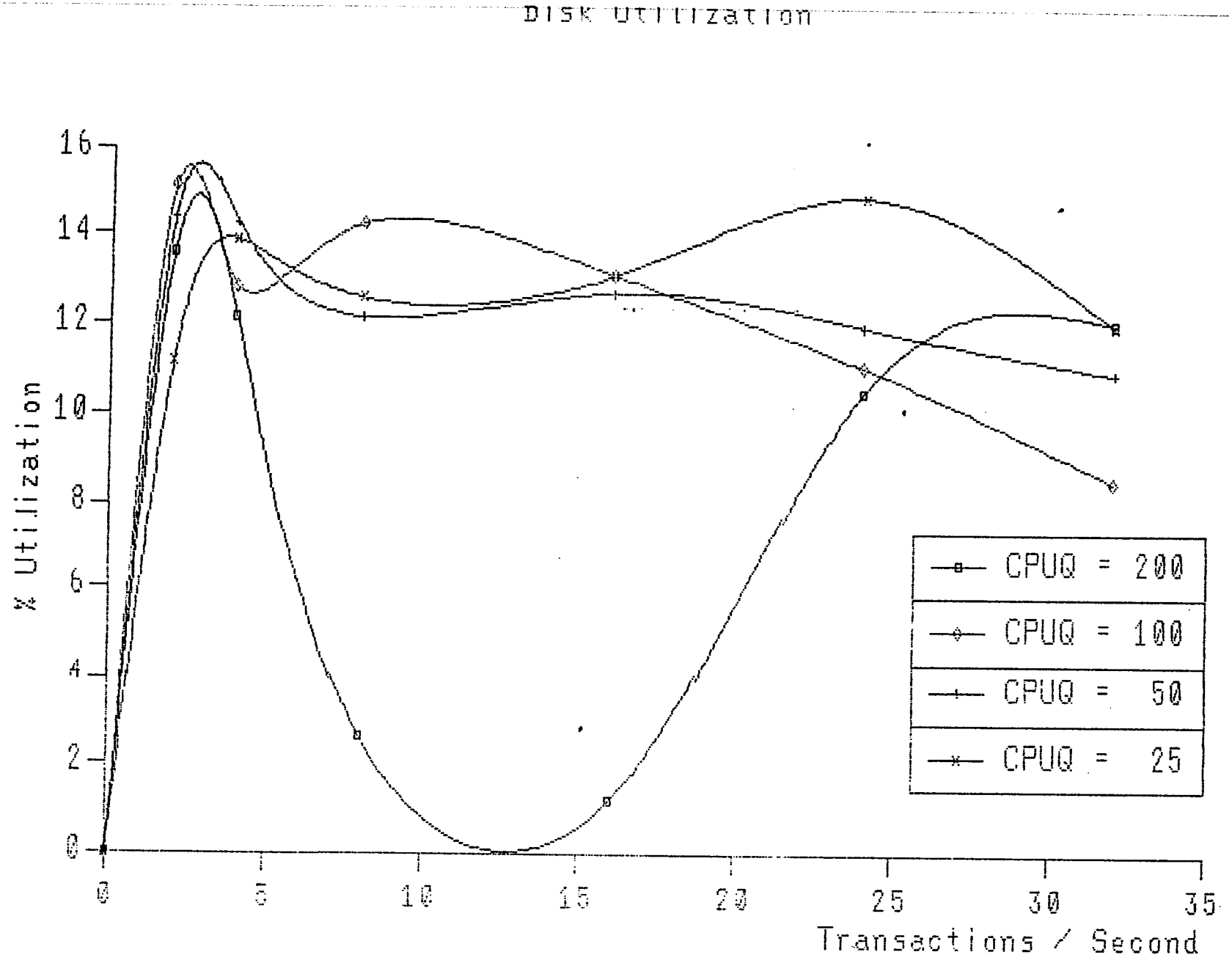


Figure 6.10 Disk Utilization

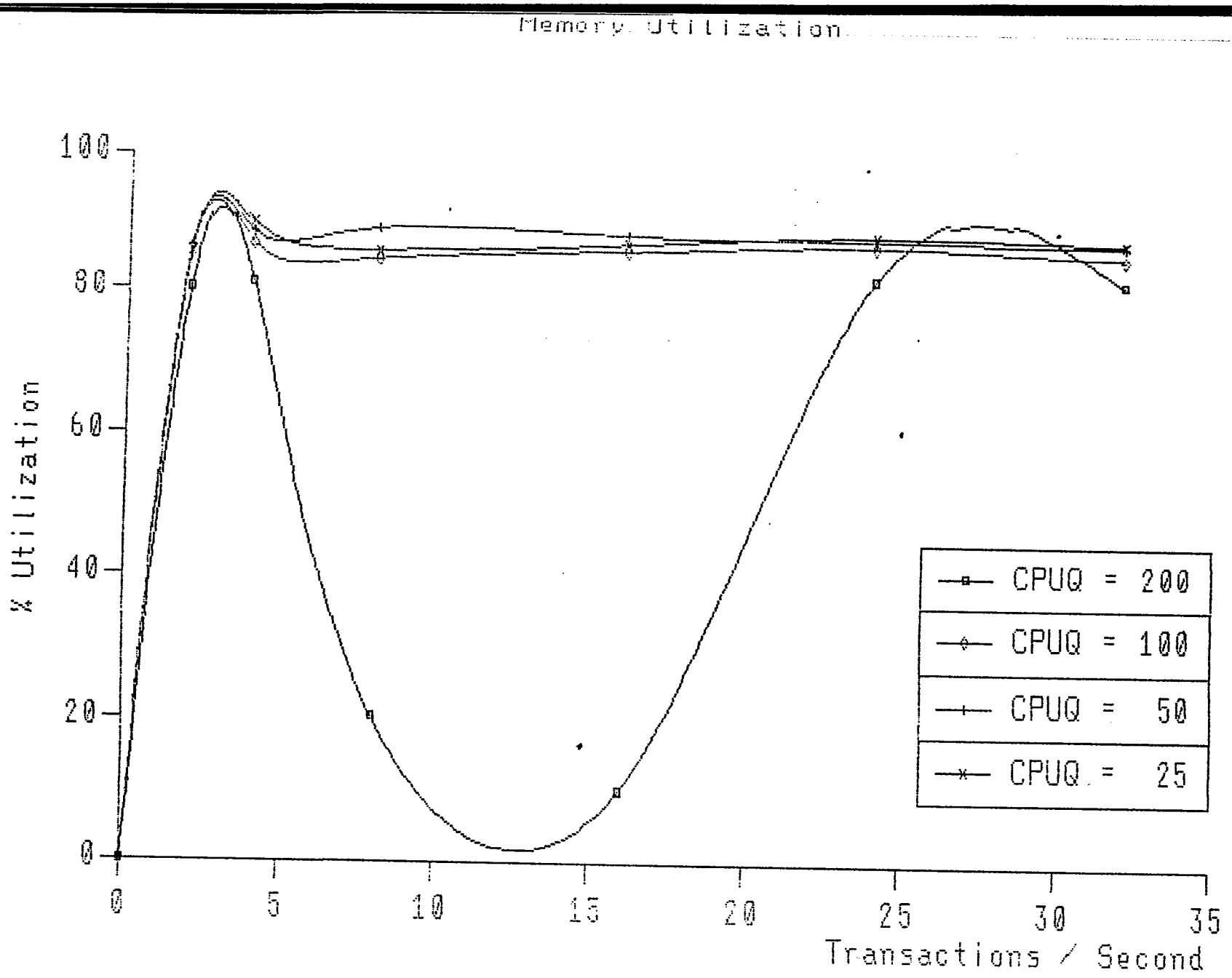


Figure 6.11 Memory Utilization

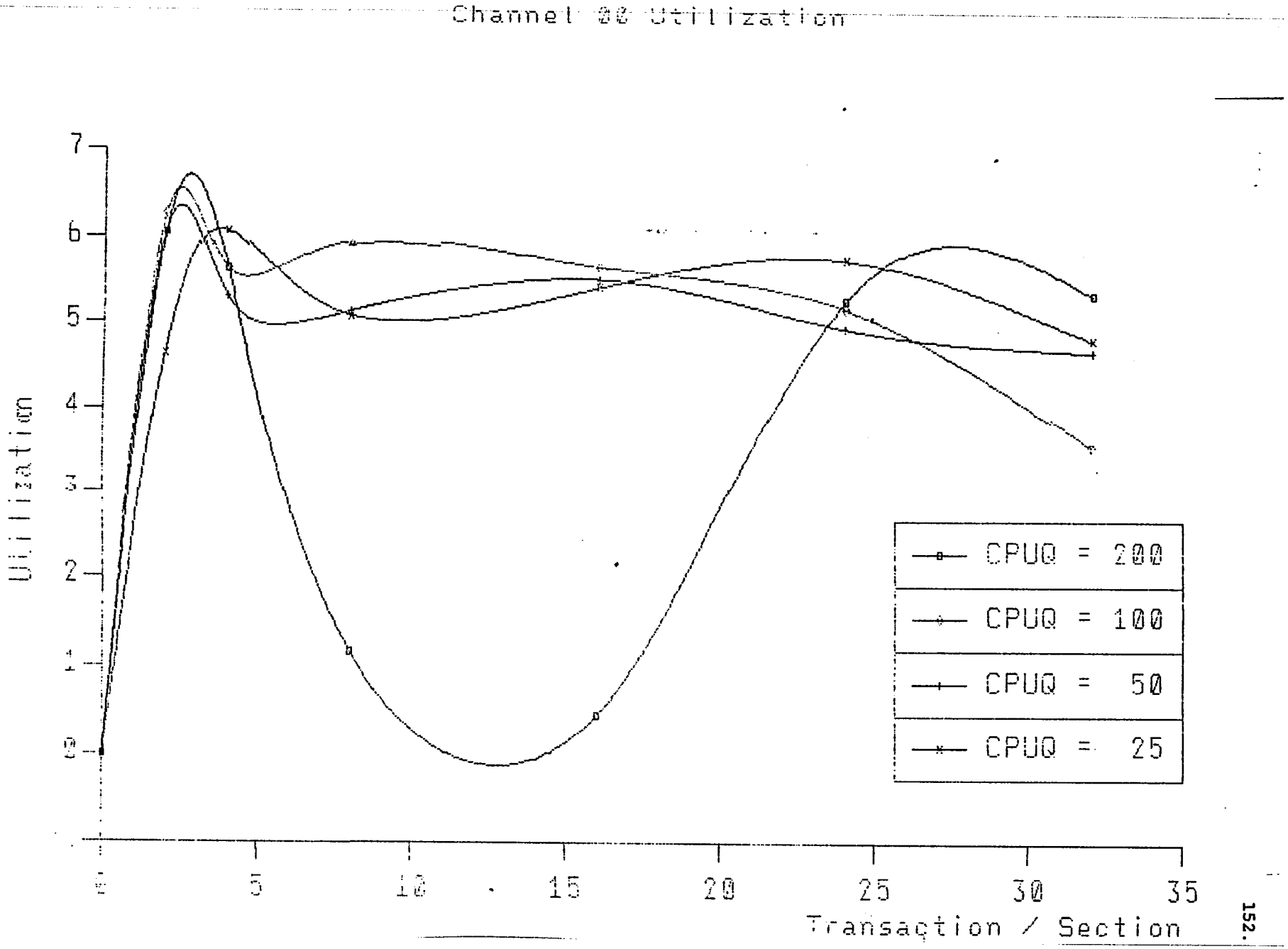


Figure 6.12 Channel 00 Utilization

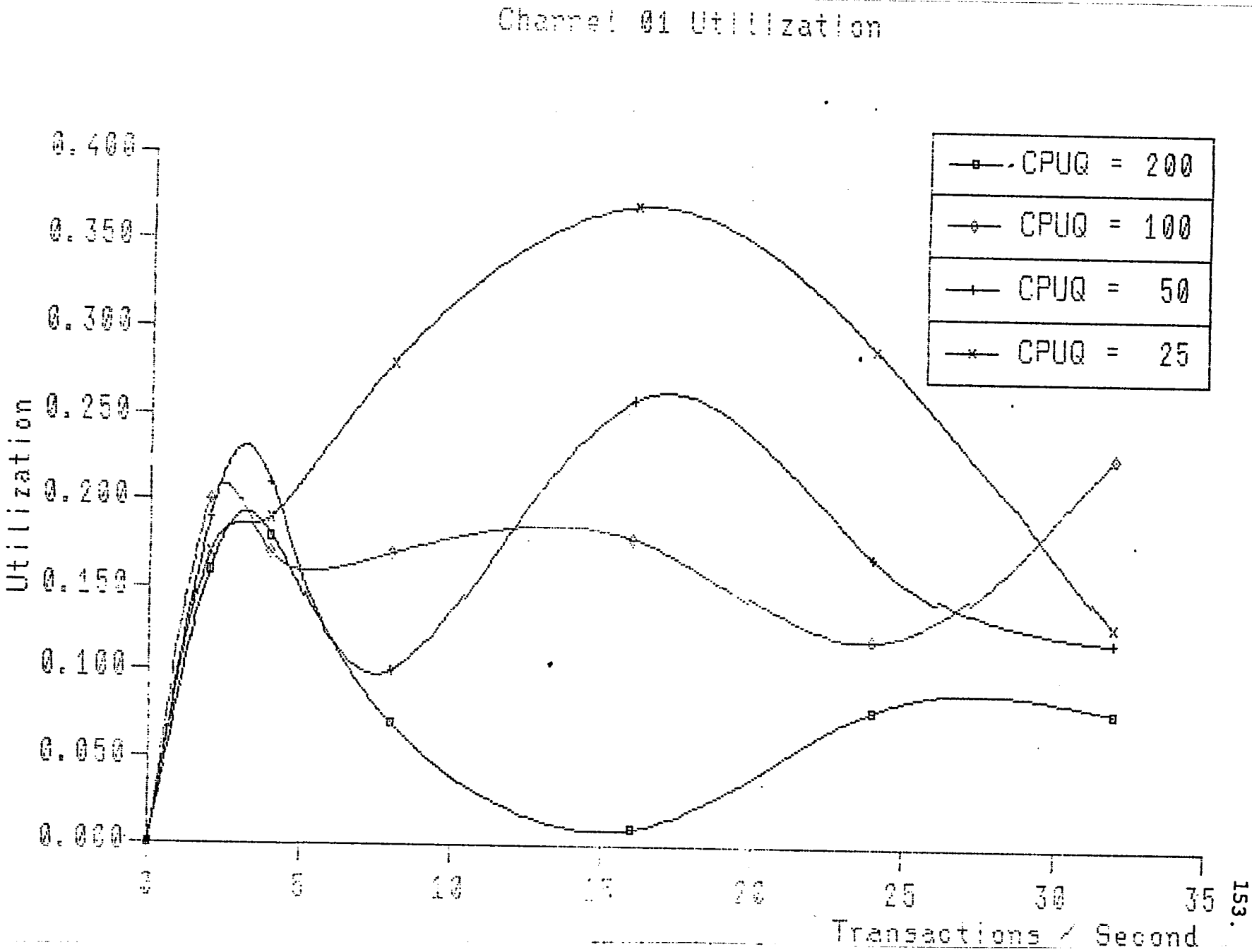


Figure 6.13 Channel 01 Utilization

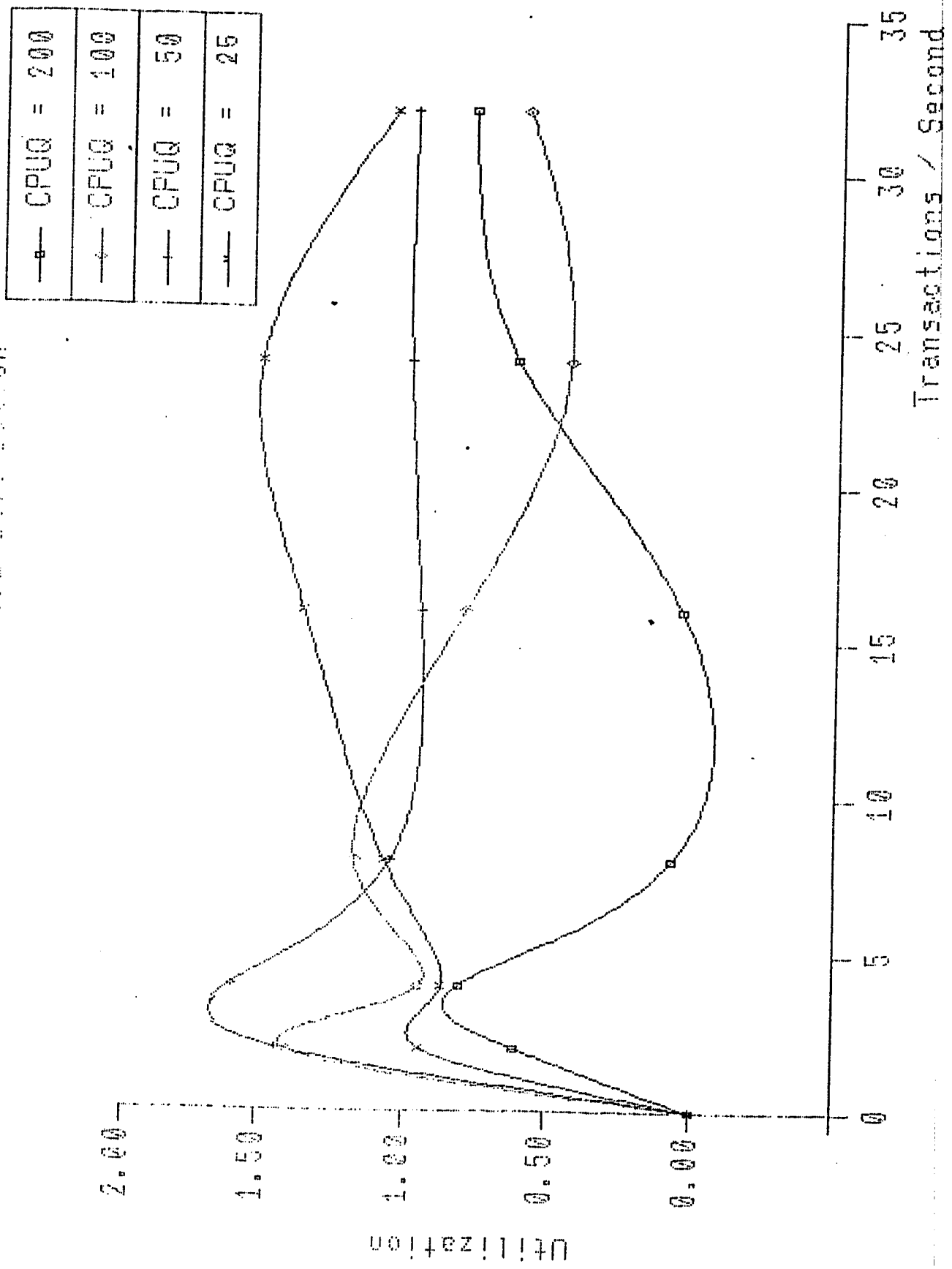


Figure 6.14 Channel 02B Utilization

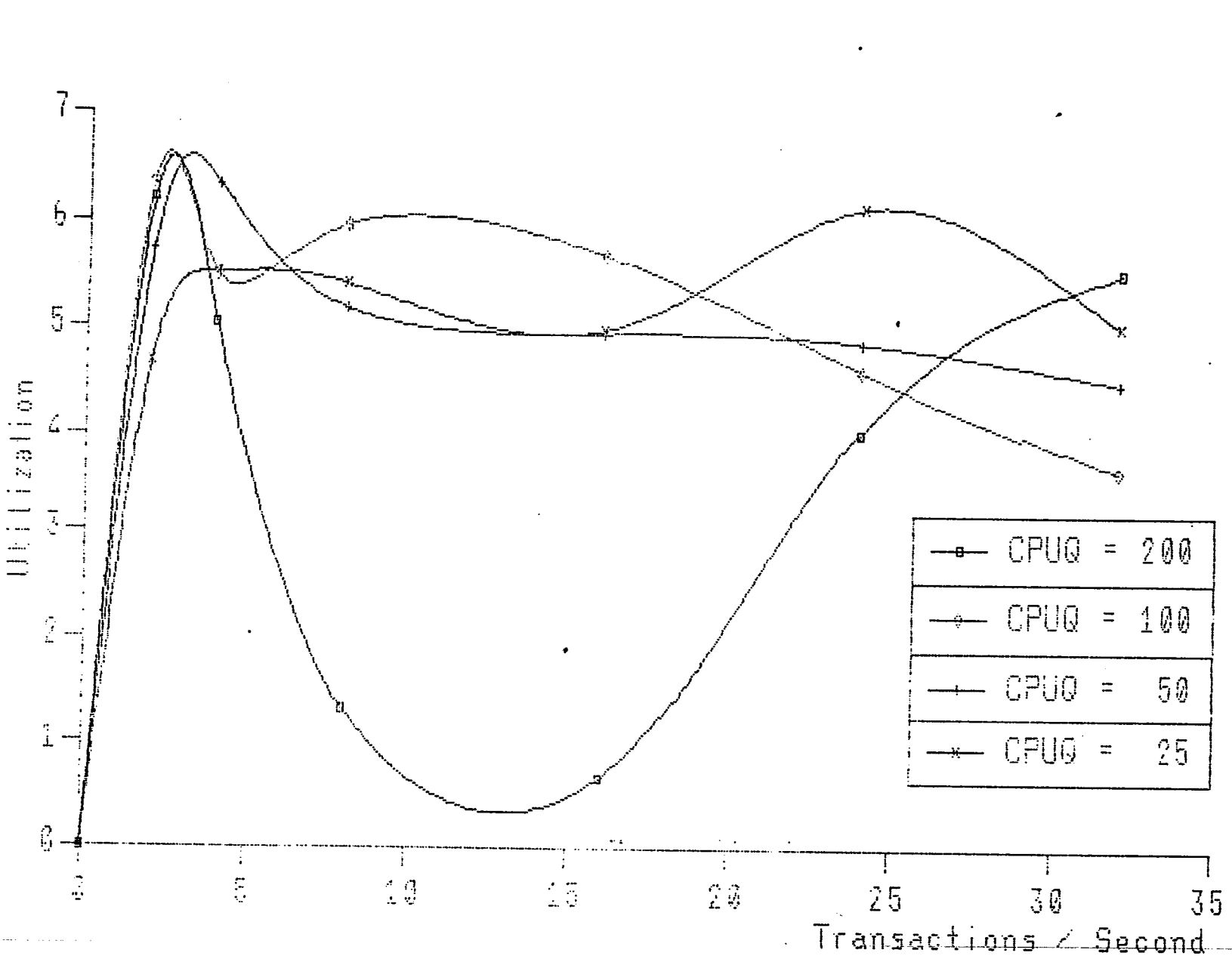


Figure 6.15 Channel 10 Utilization

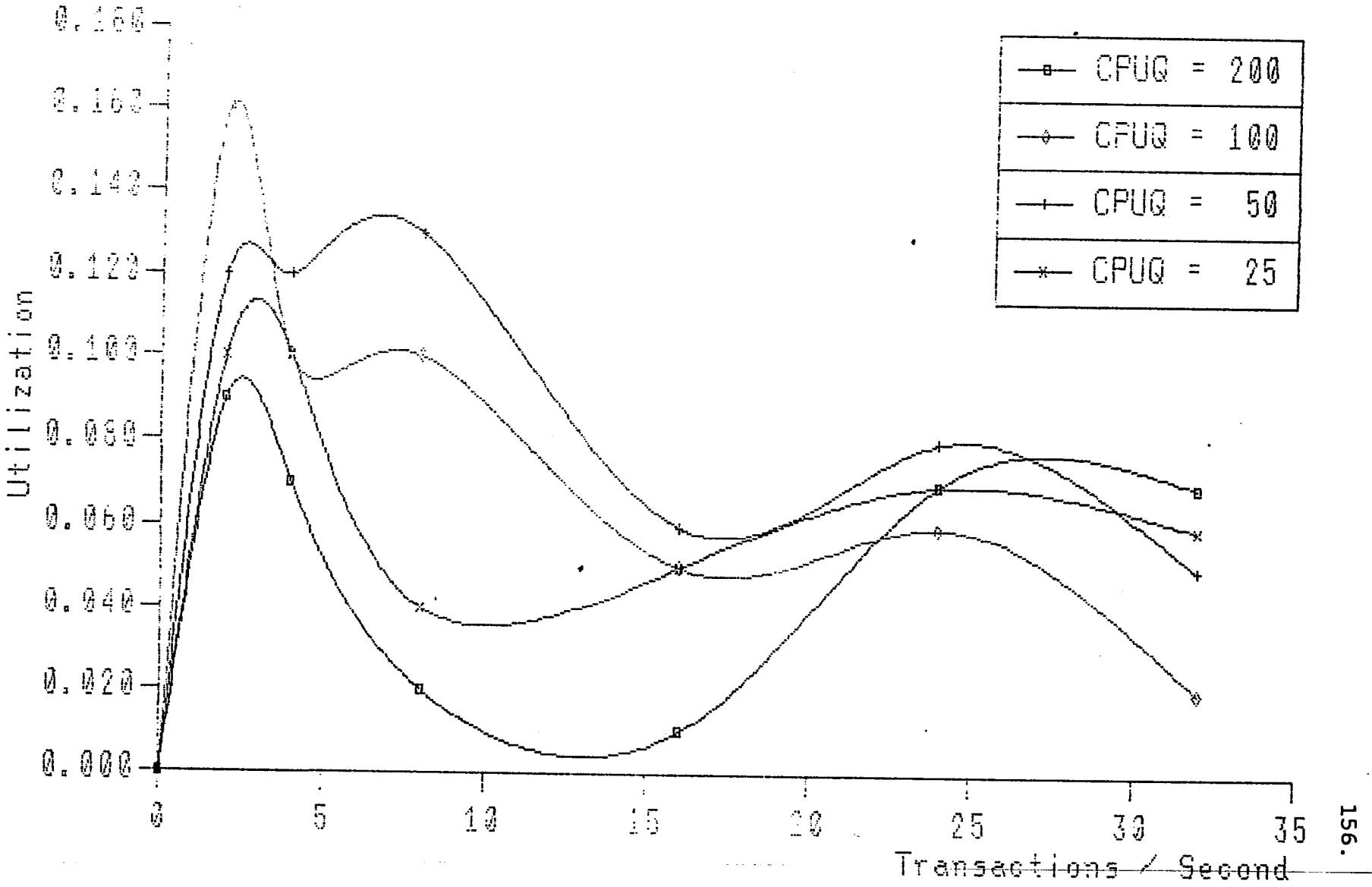


Figure 6.16 Channel 11 Utilization

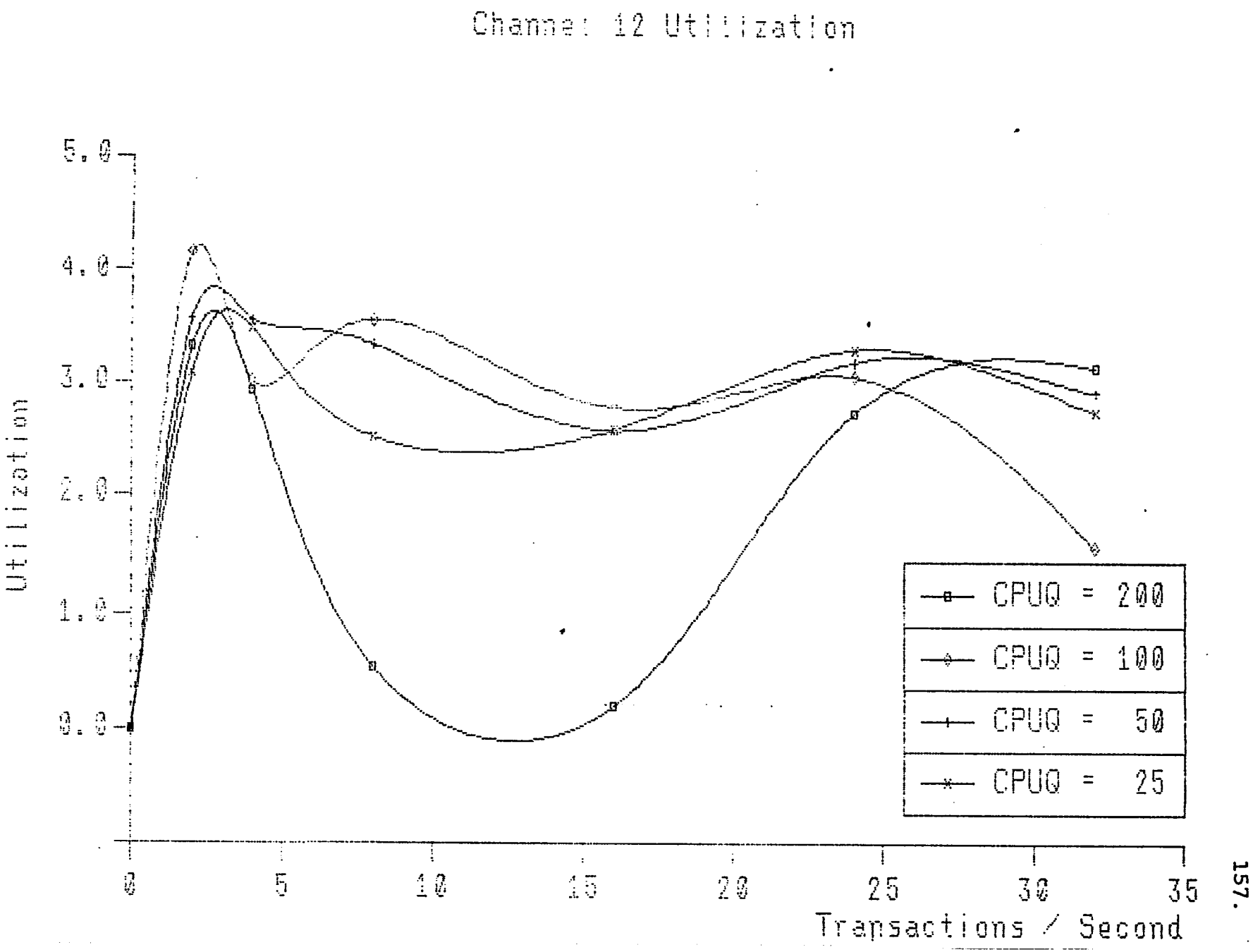


Figure 6.17 Channel 12 Utilization

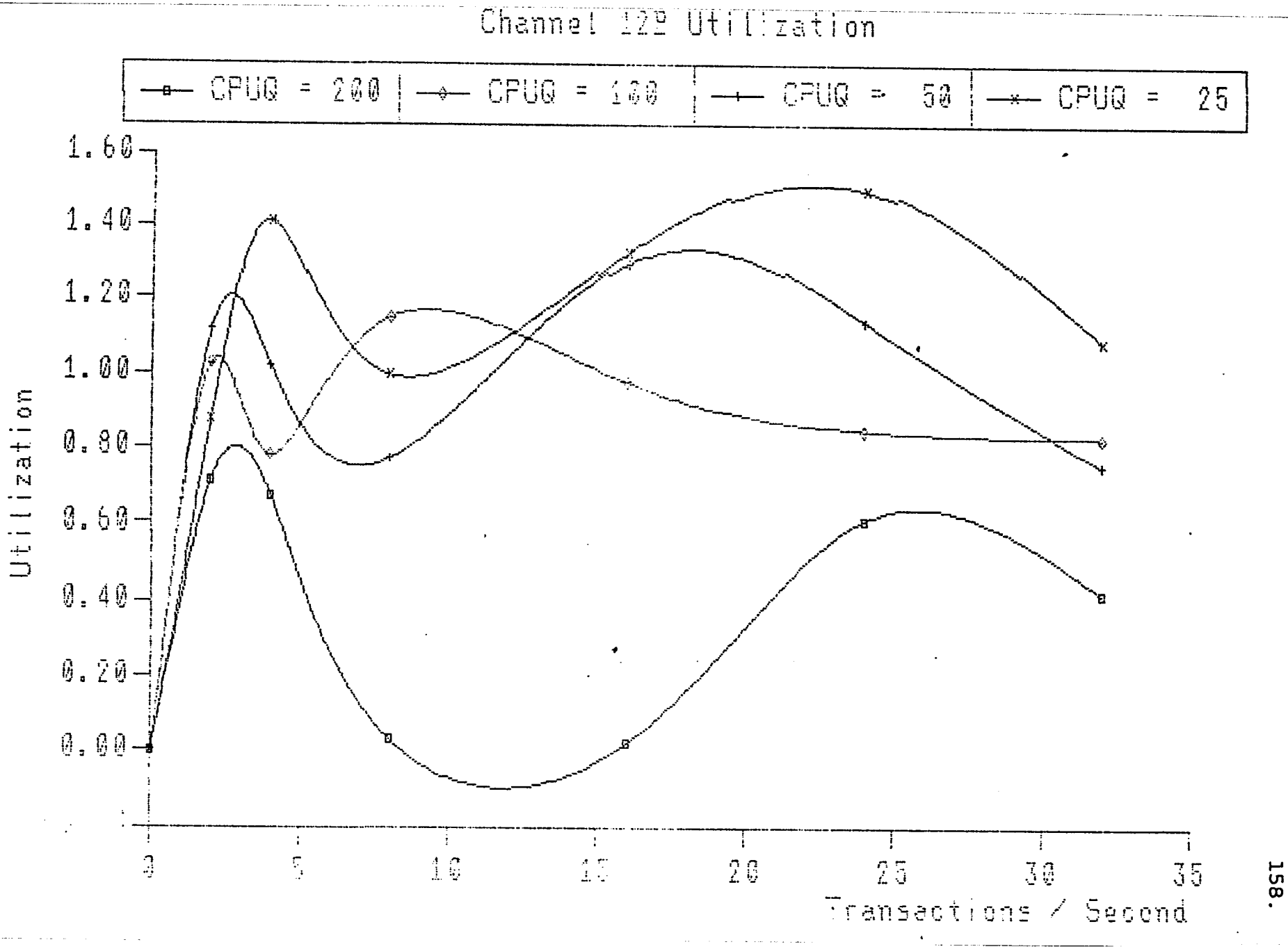


Figure 6.18 Channel 12B Utilization

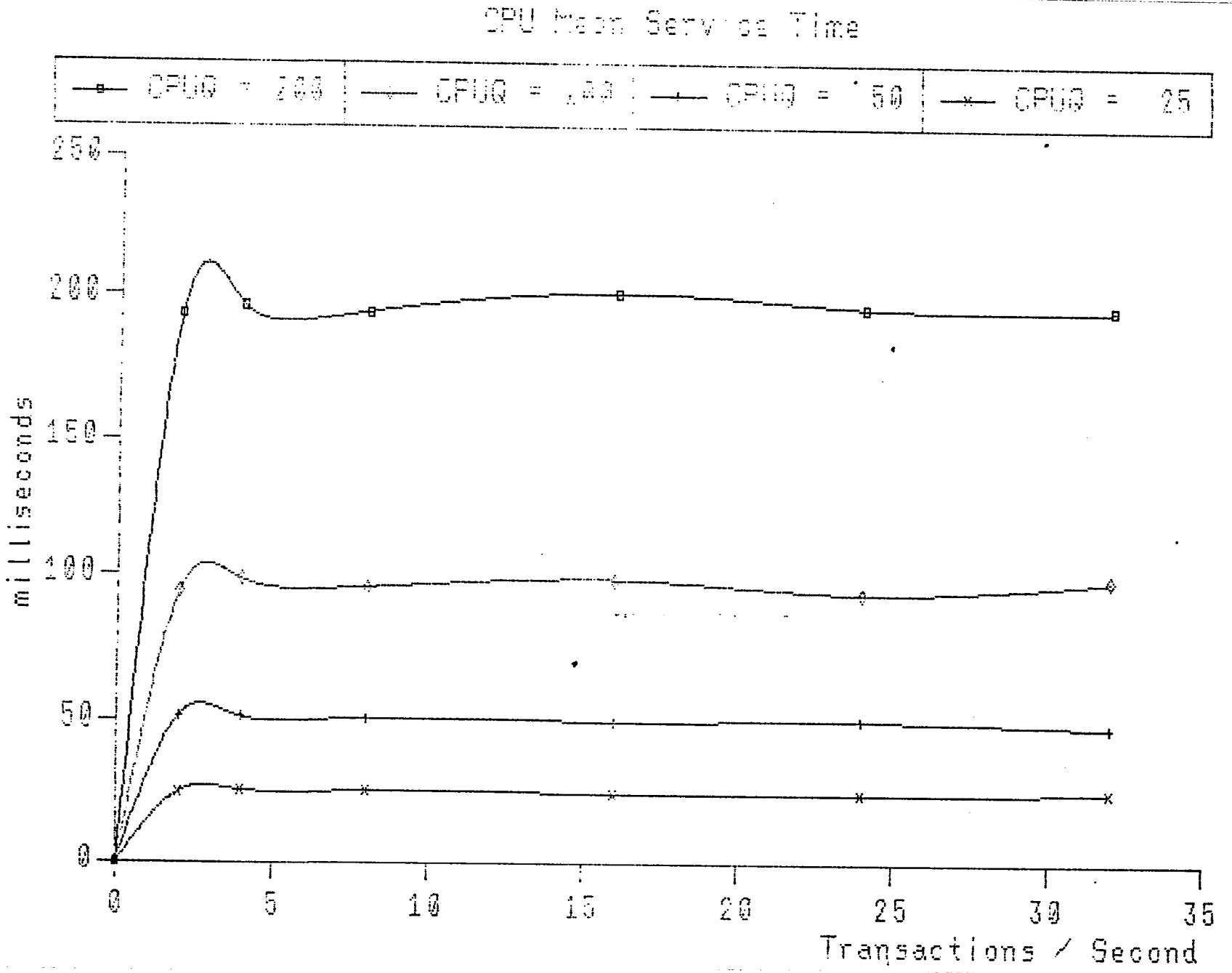


Figure 6.19 CPU Mean Service Time

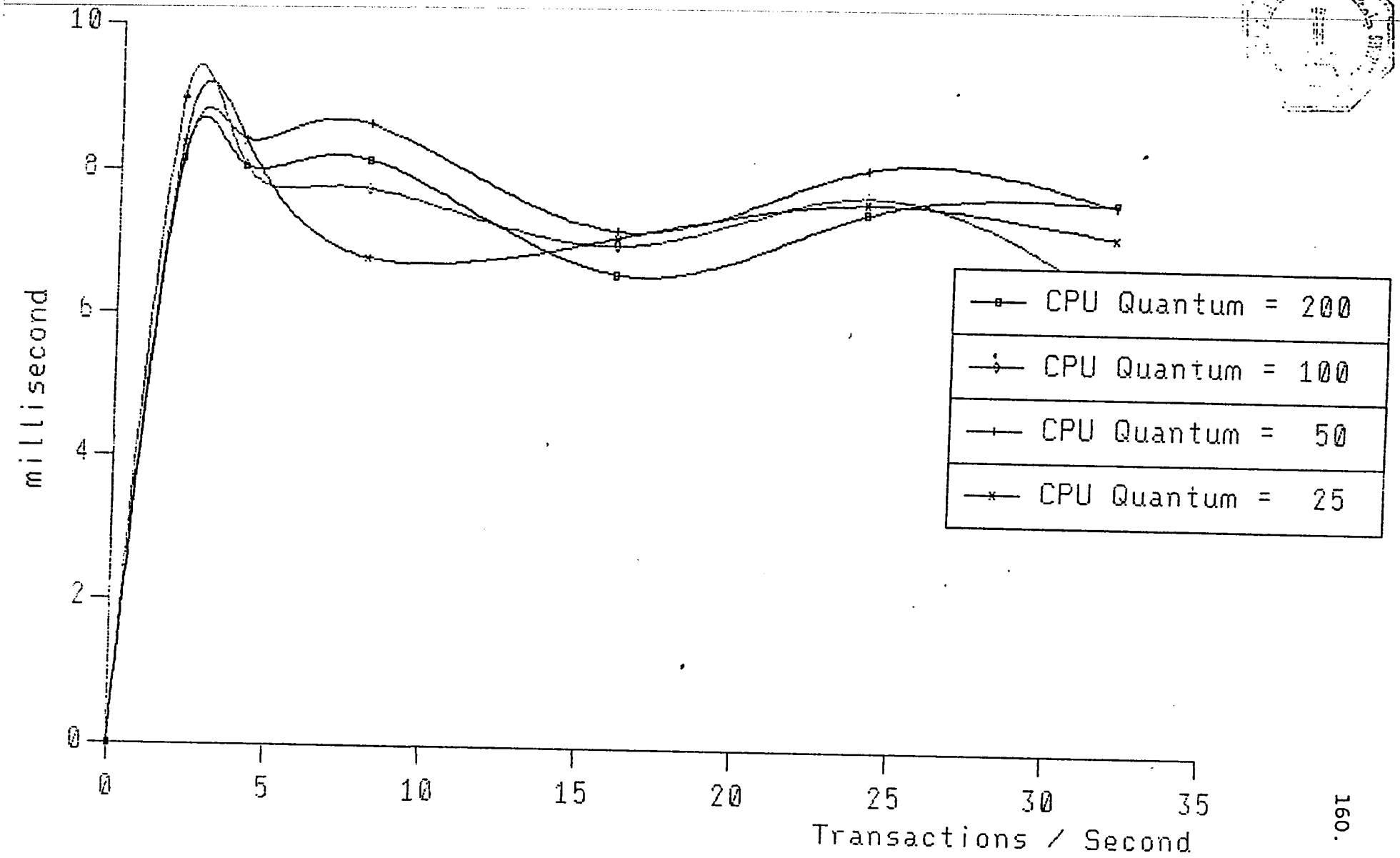


Figure 6.20 Drum Mean Service Time

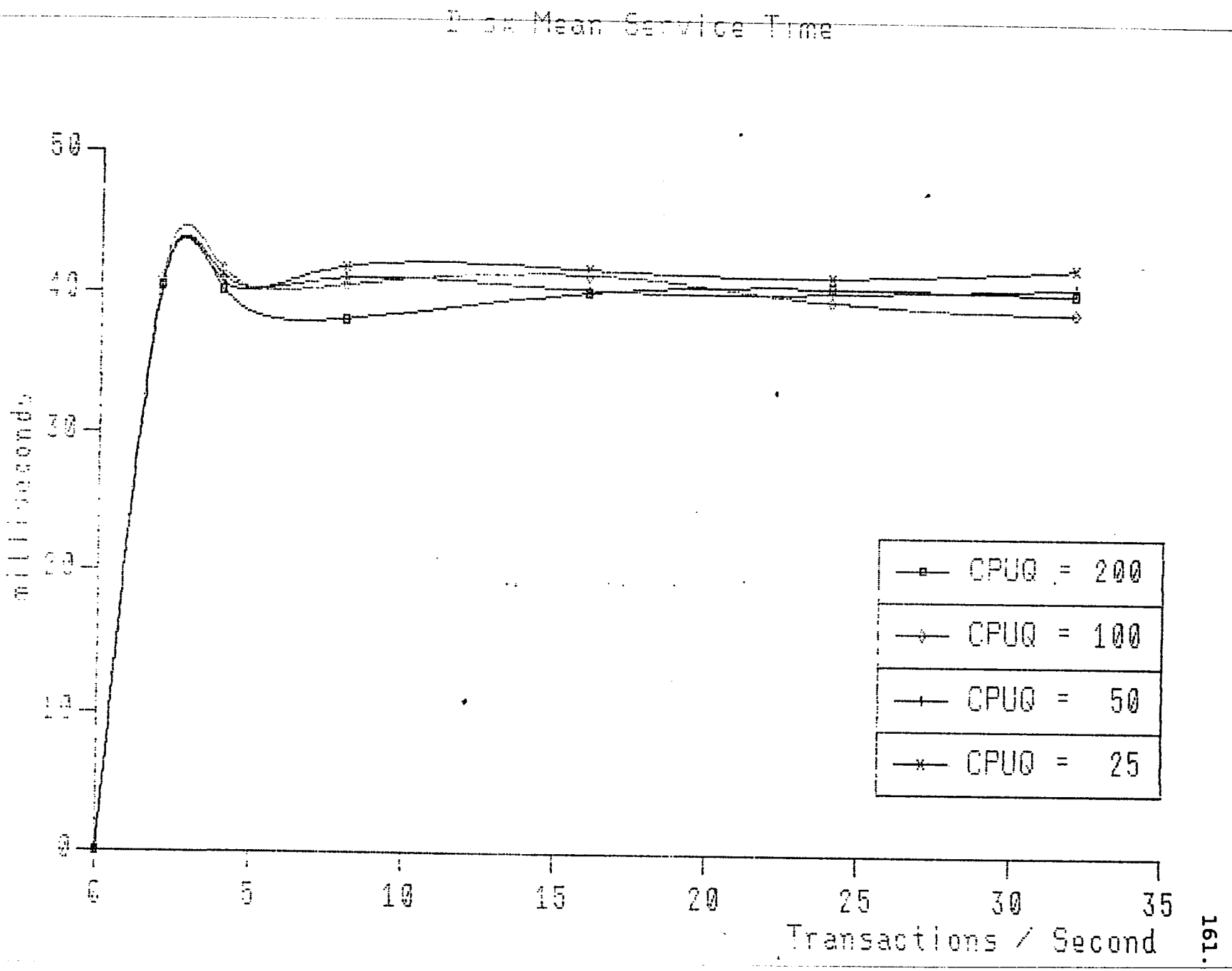


Figure 6.21 Disk Mean Service Time

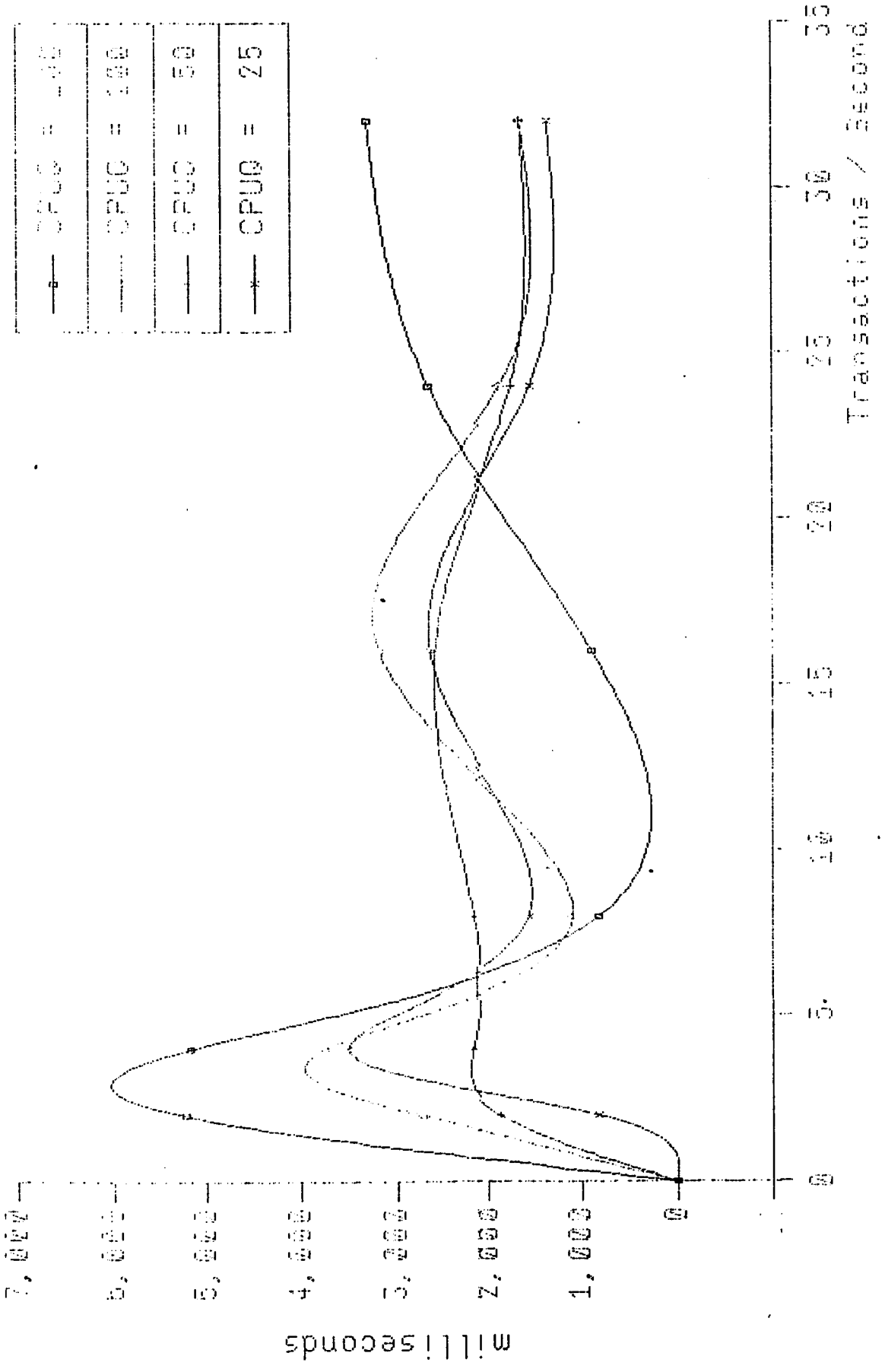


Figure 6.22 Mean Response Time

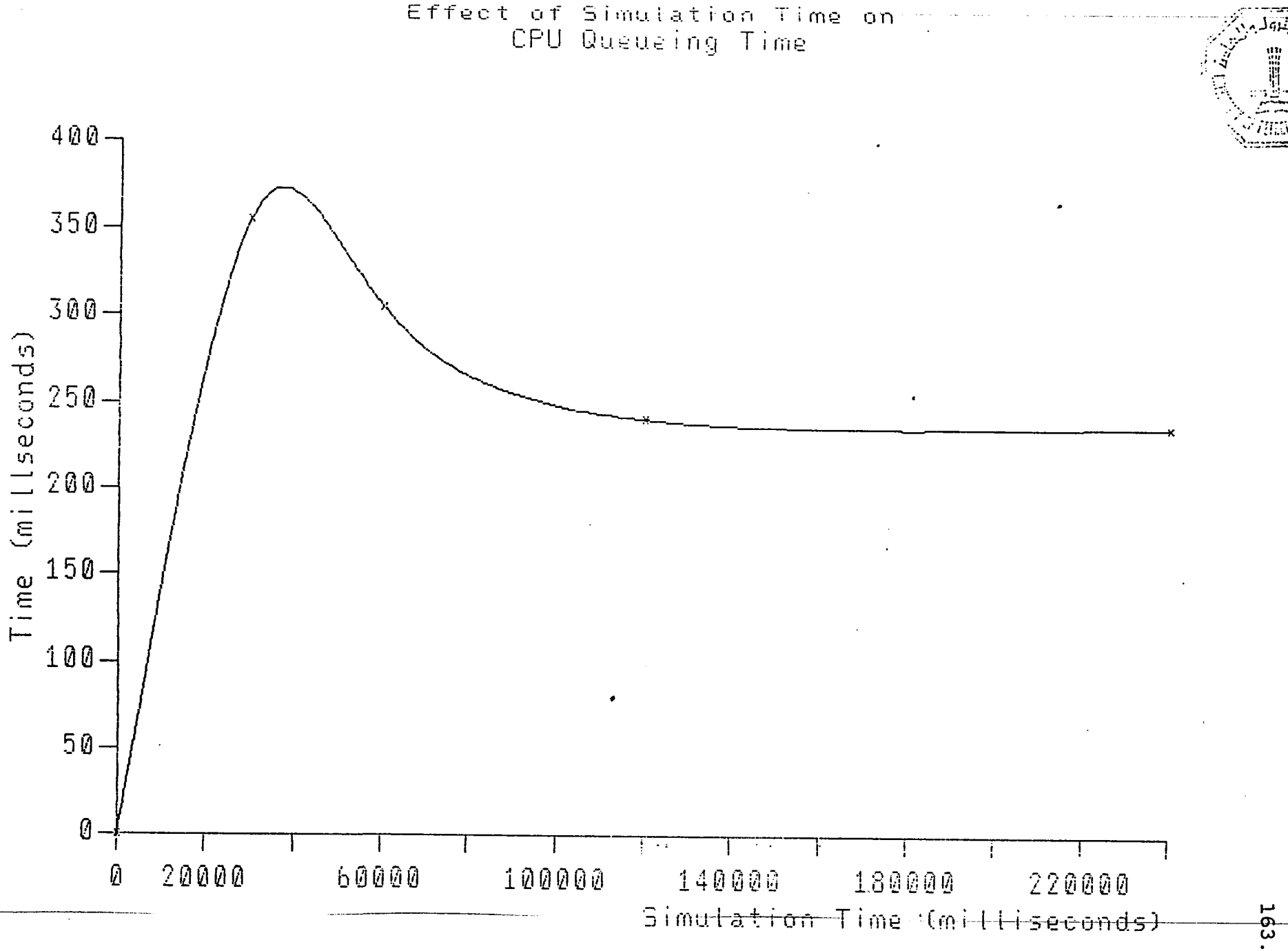


Figure 6.23 Effect of simulation time on CPU Queueing Time

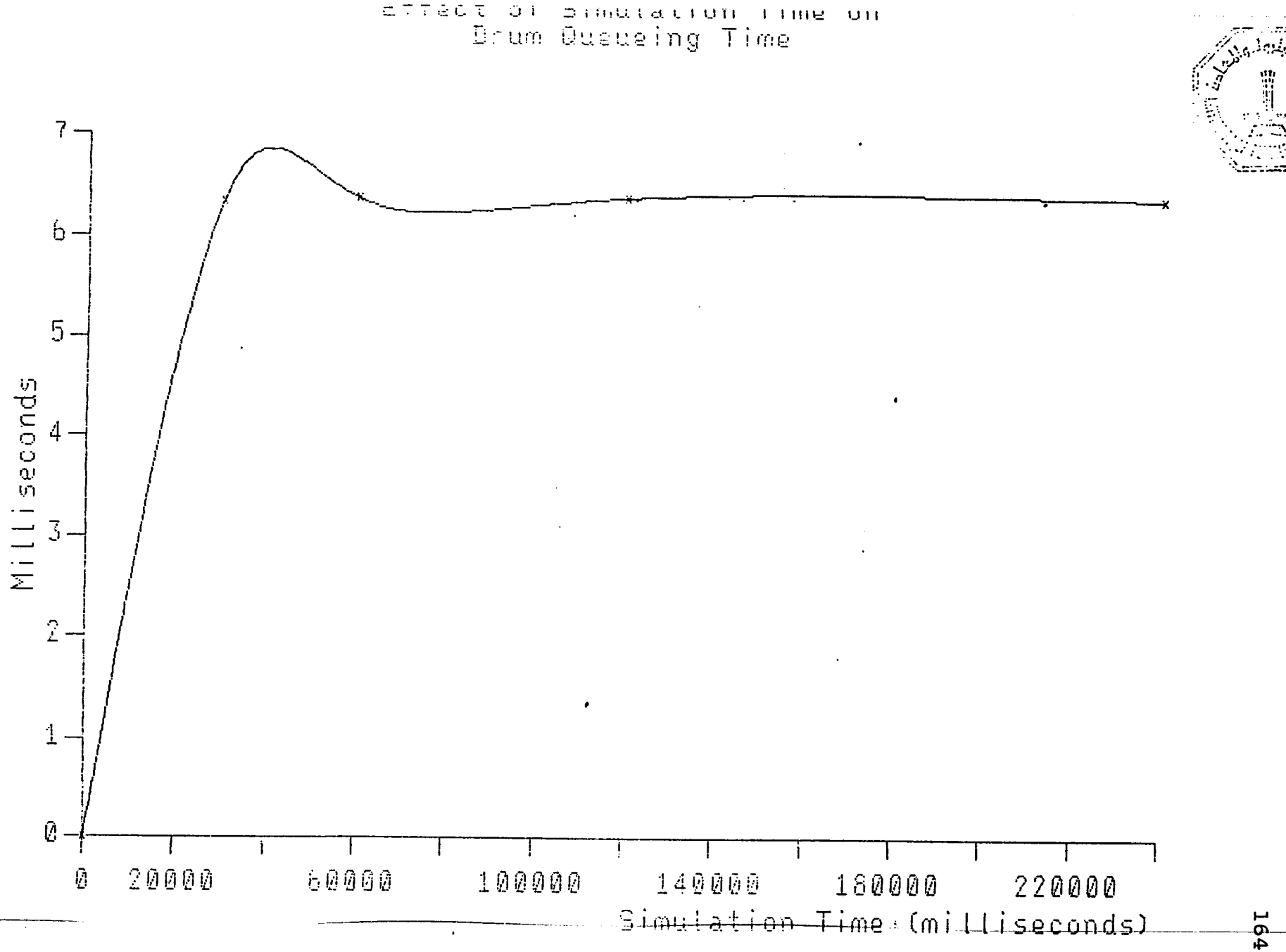


Figure 6.24 Effect of simulation time on Drum Queueing Time

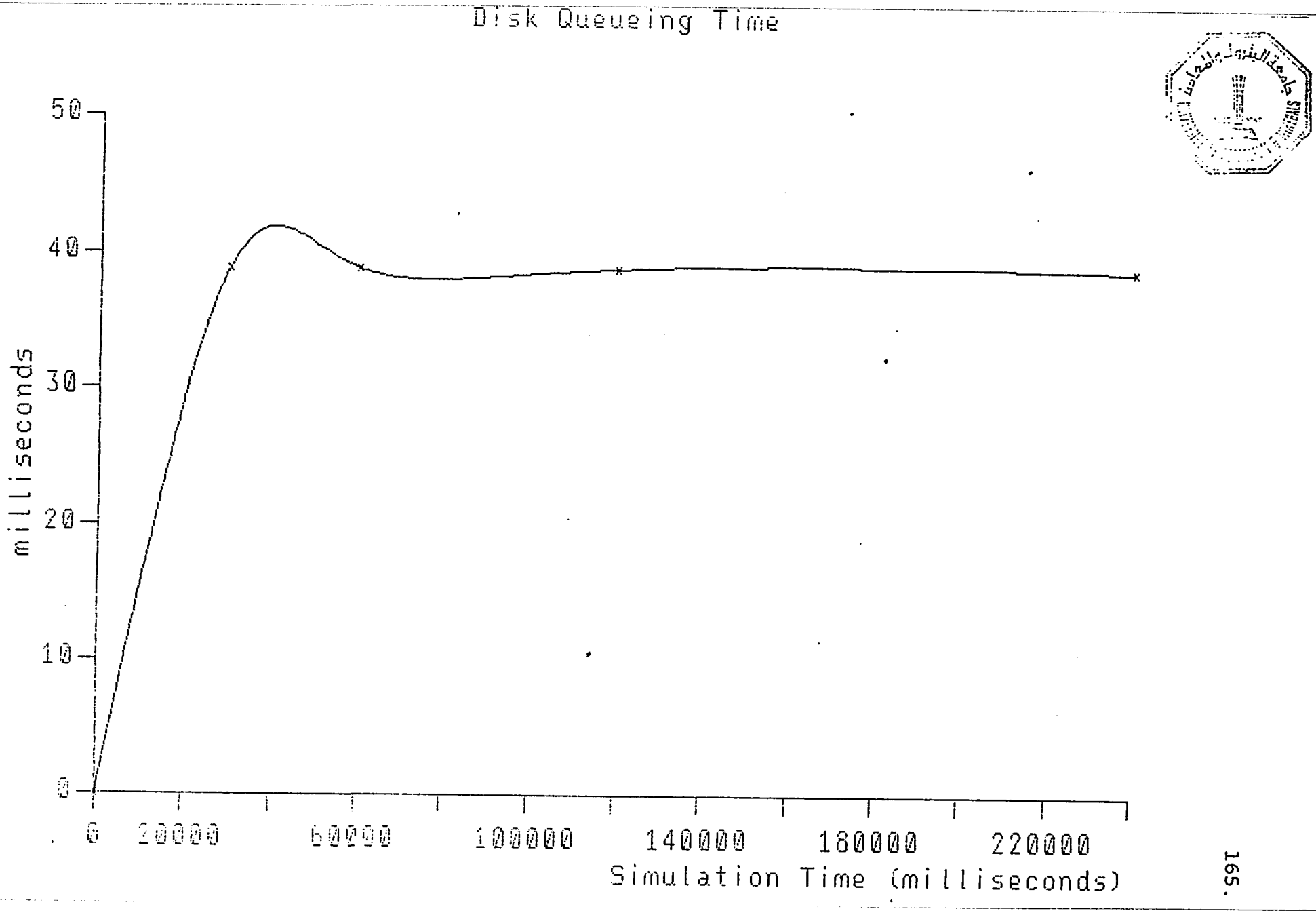


Figure 6.25 Effect of simulation time on Disk Queueing Time

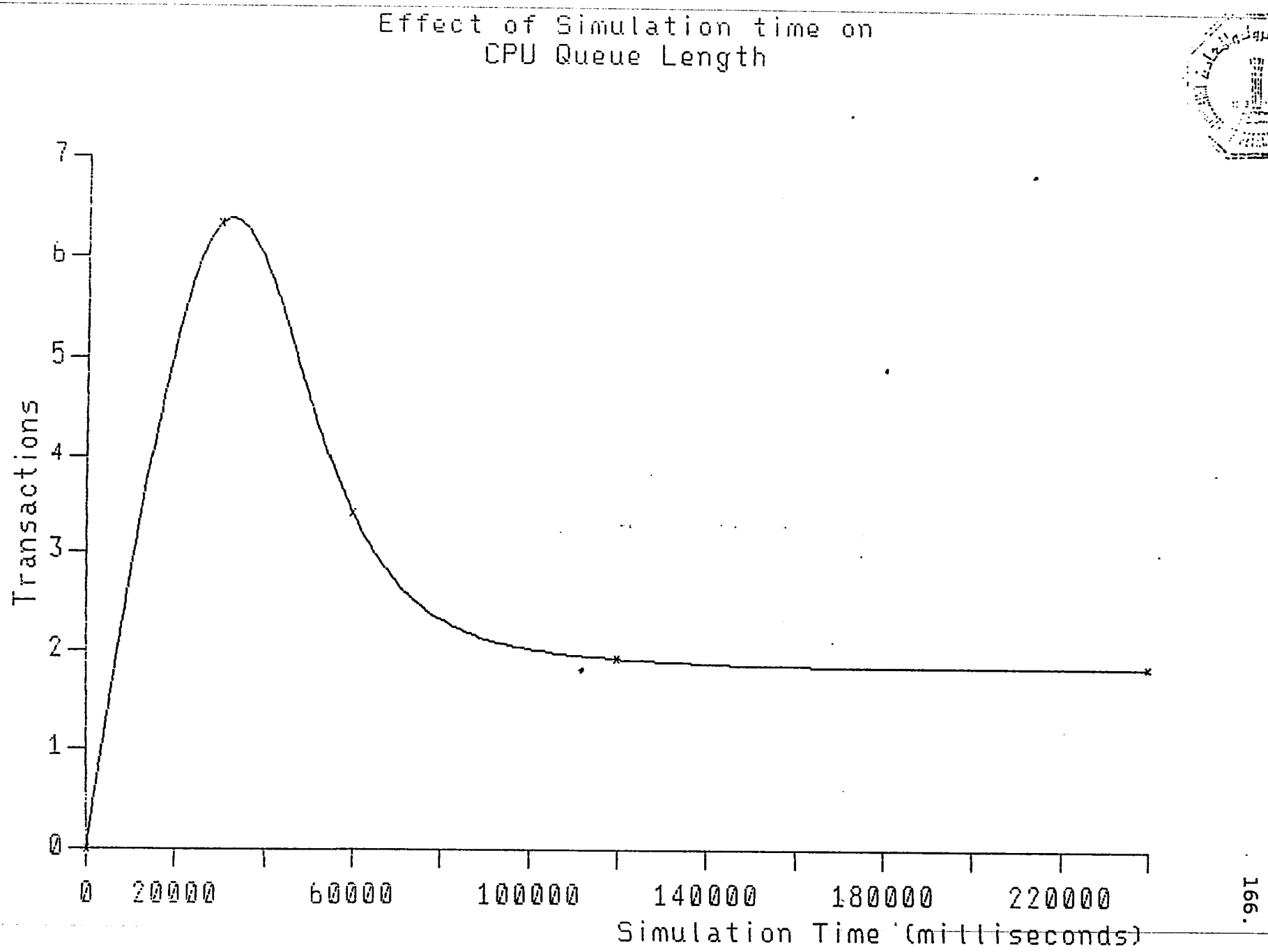


Figure 6.26 Effect of simulation time on CPU Queue Length

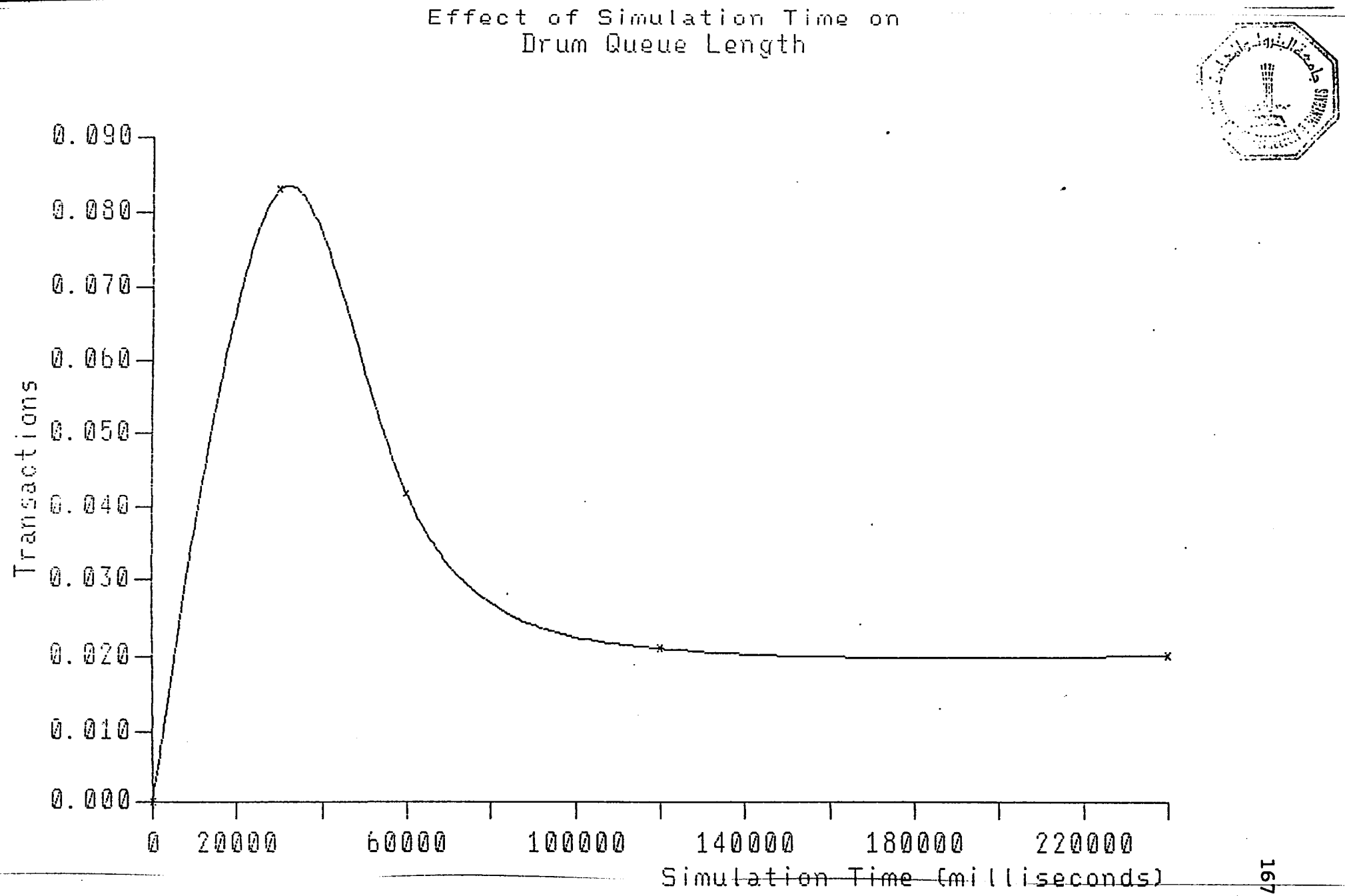


Figure 6.27 Effect of simulation time on Drum Queue Length

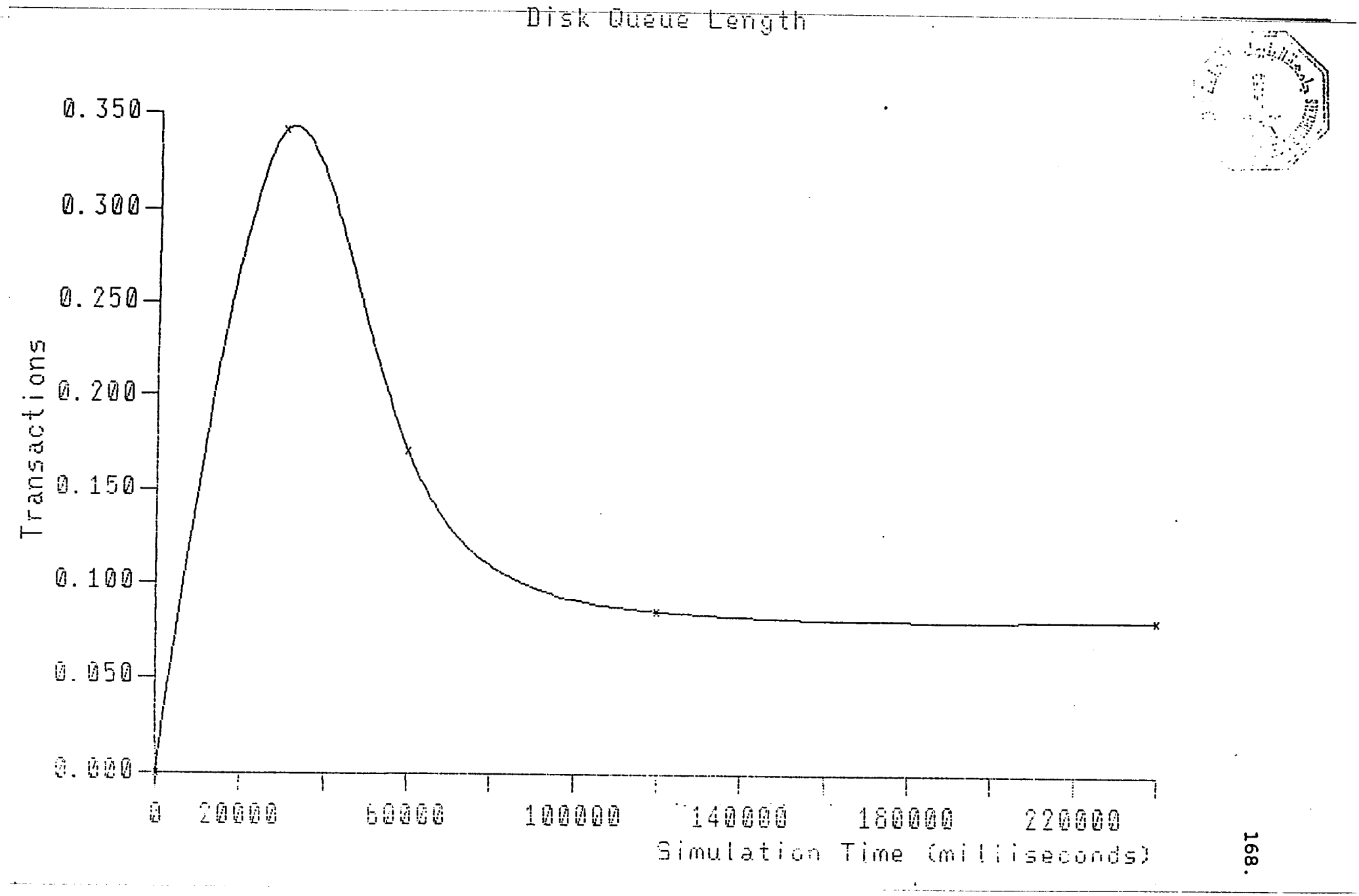


Figure 6.28 Effect of simulation time on Disk Queue Length

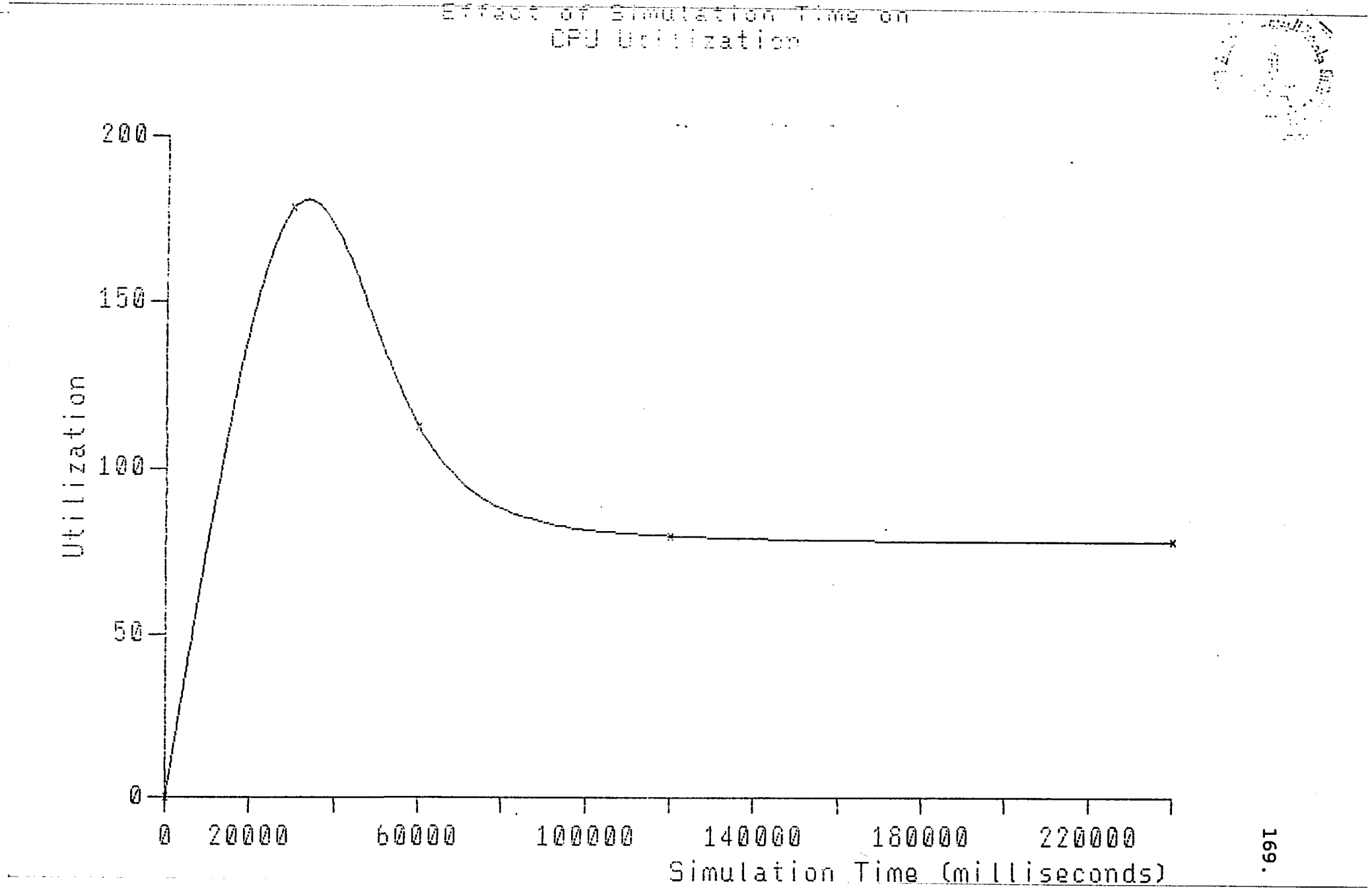


Figure 6.29 Effect of simulation time on CPU Utilization

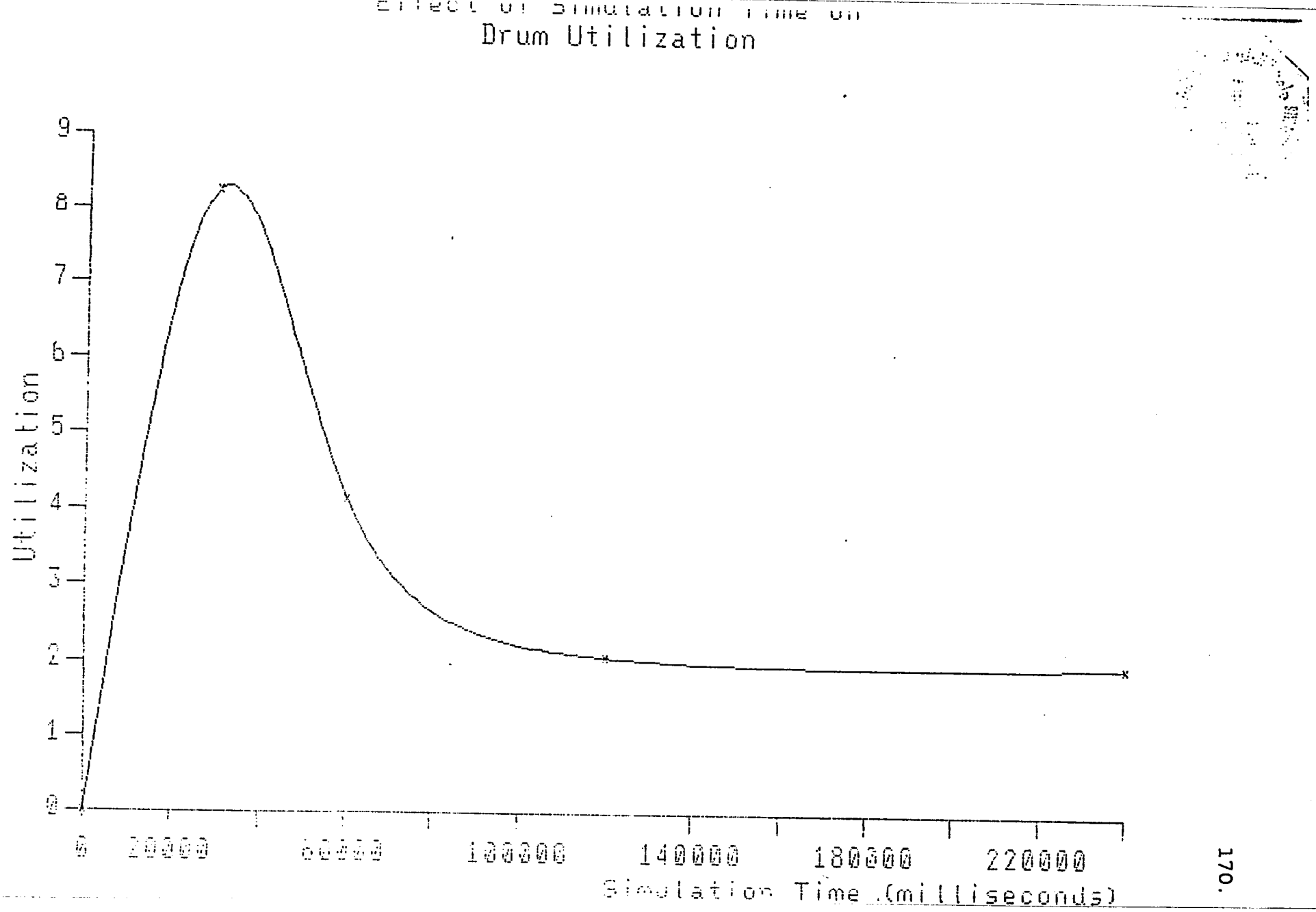


Figure 6.30 Effect of simulation time on Drum Utilization

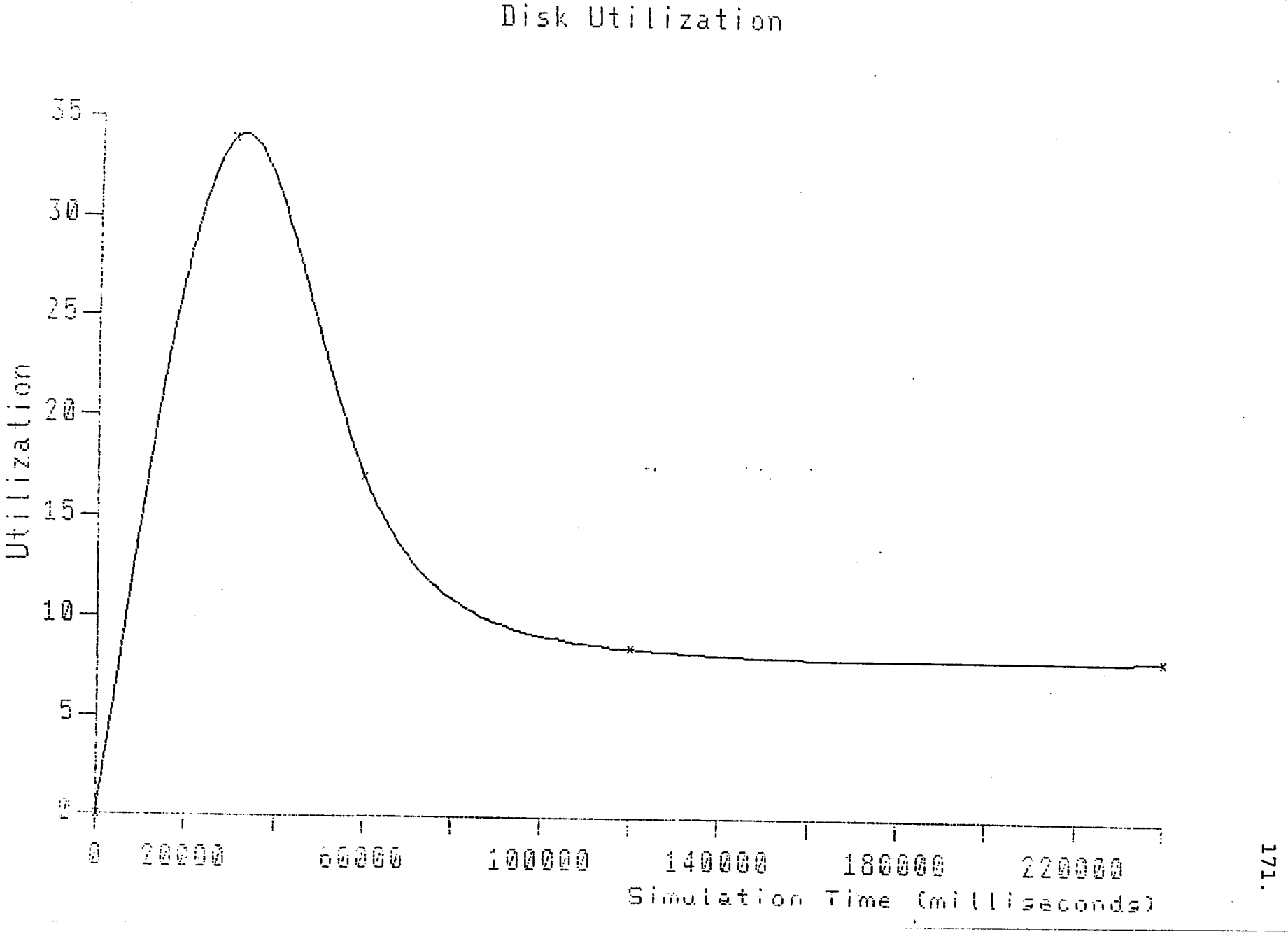


Figure 6.31 Effect of simulation time on Disk Utilization

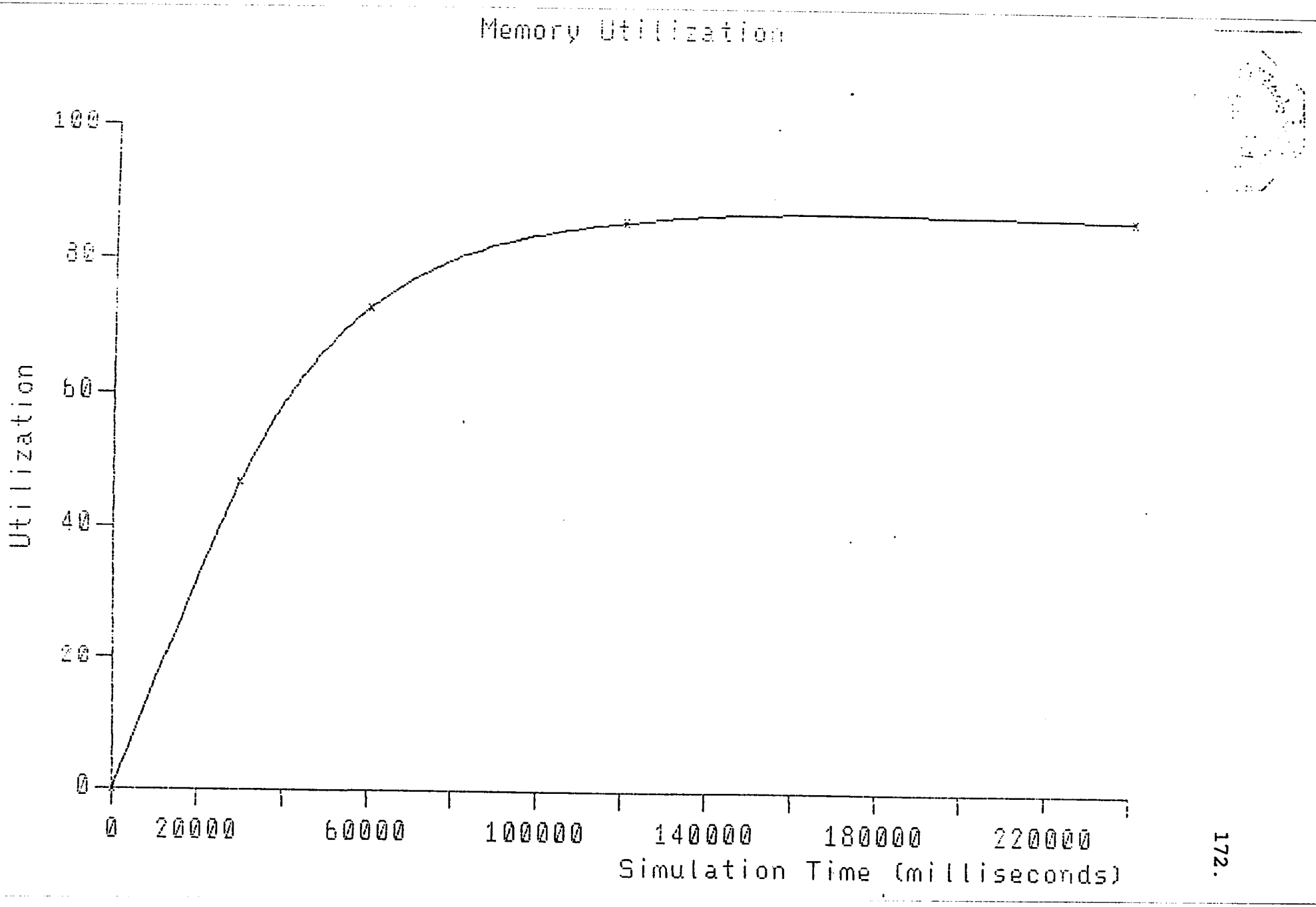


Figure 6.32 Effect of simulation time on Memory Utilization

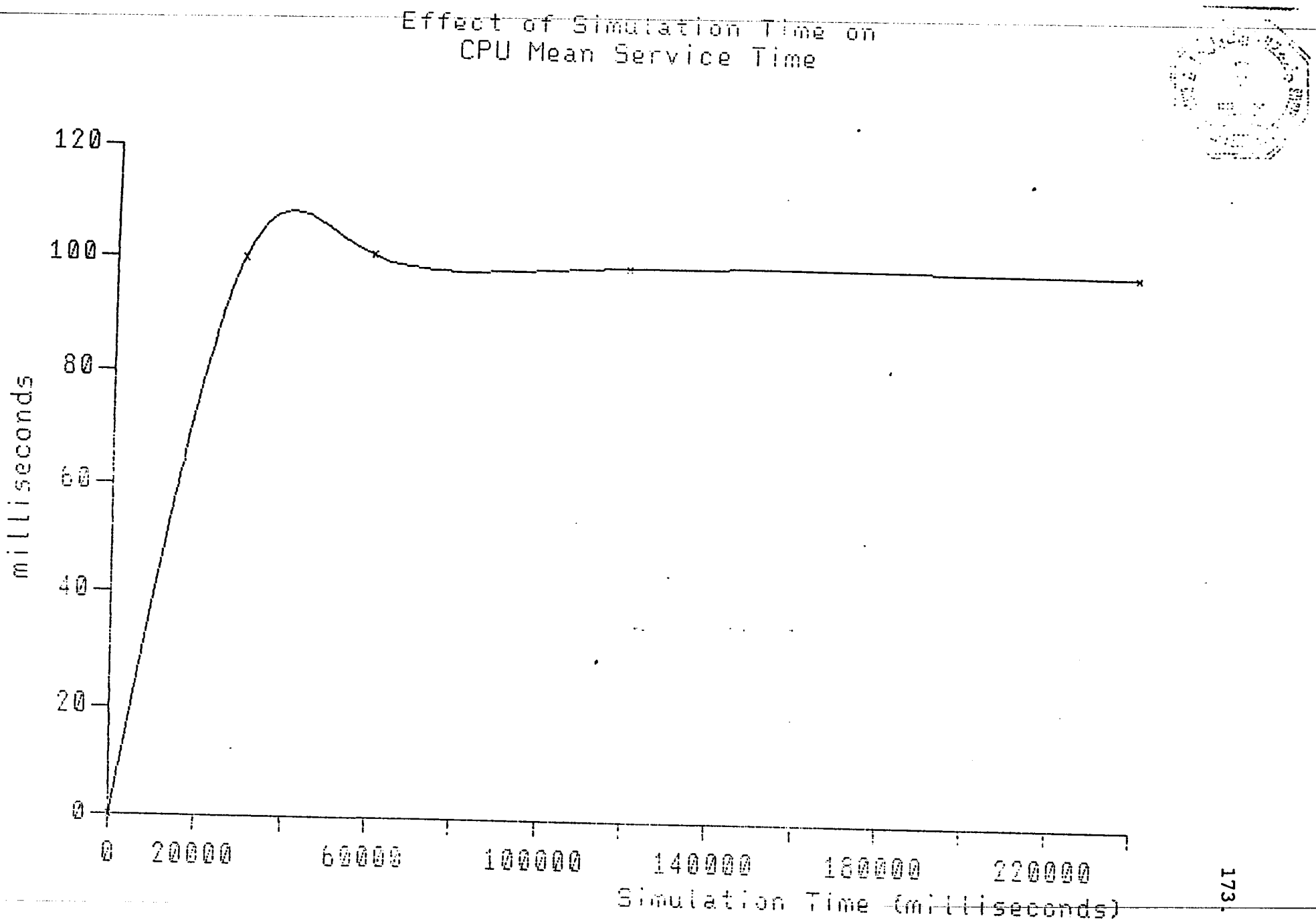


Figure 6.33 Effect of simulation time on CPU Mean Service Time

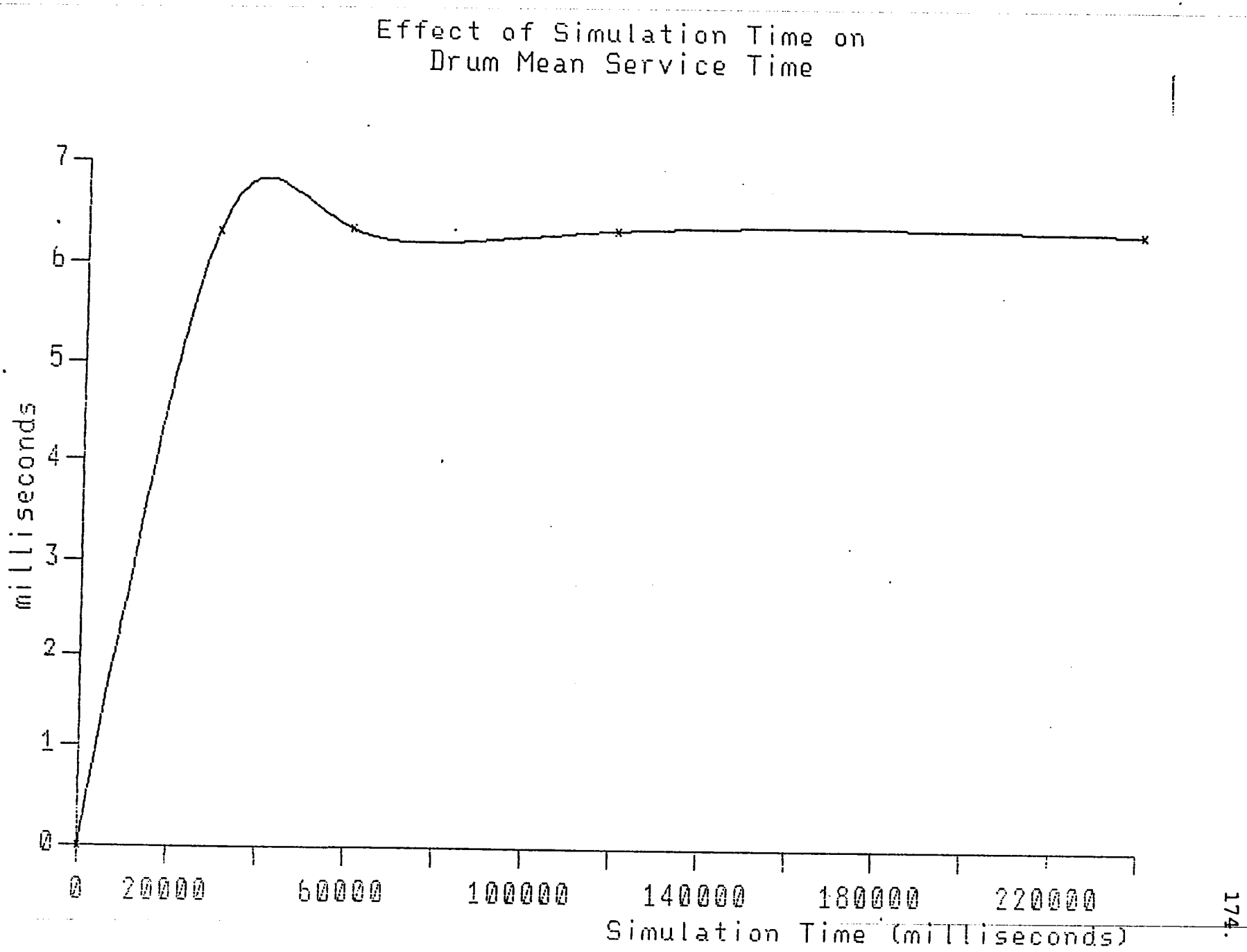


Figure 6.34 Effect of simulation time on Drum Mean Service Time

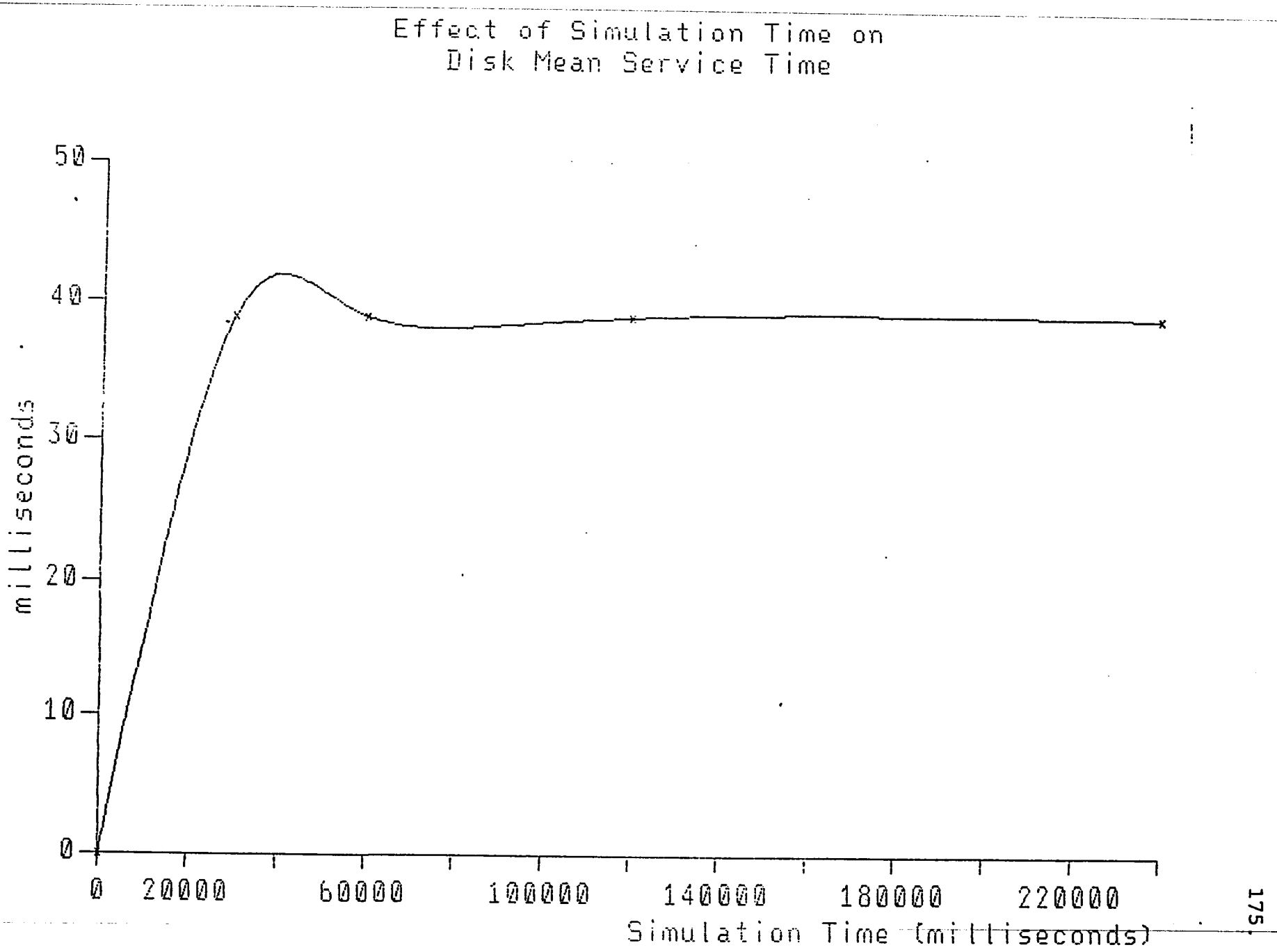


Figure 6.35 Effect of simulation time on Disk Mean Service Time

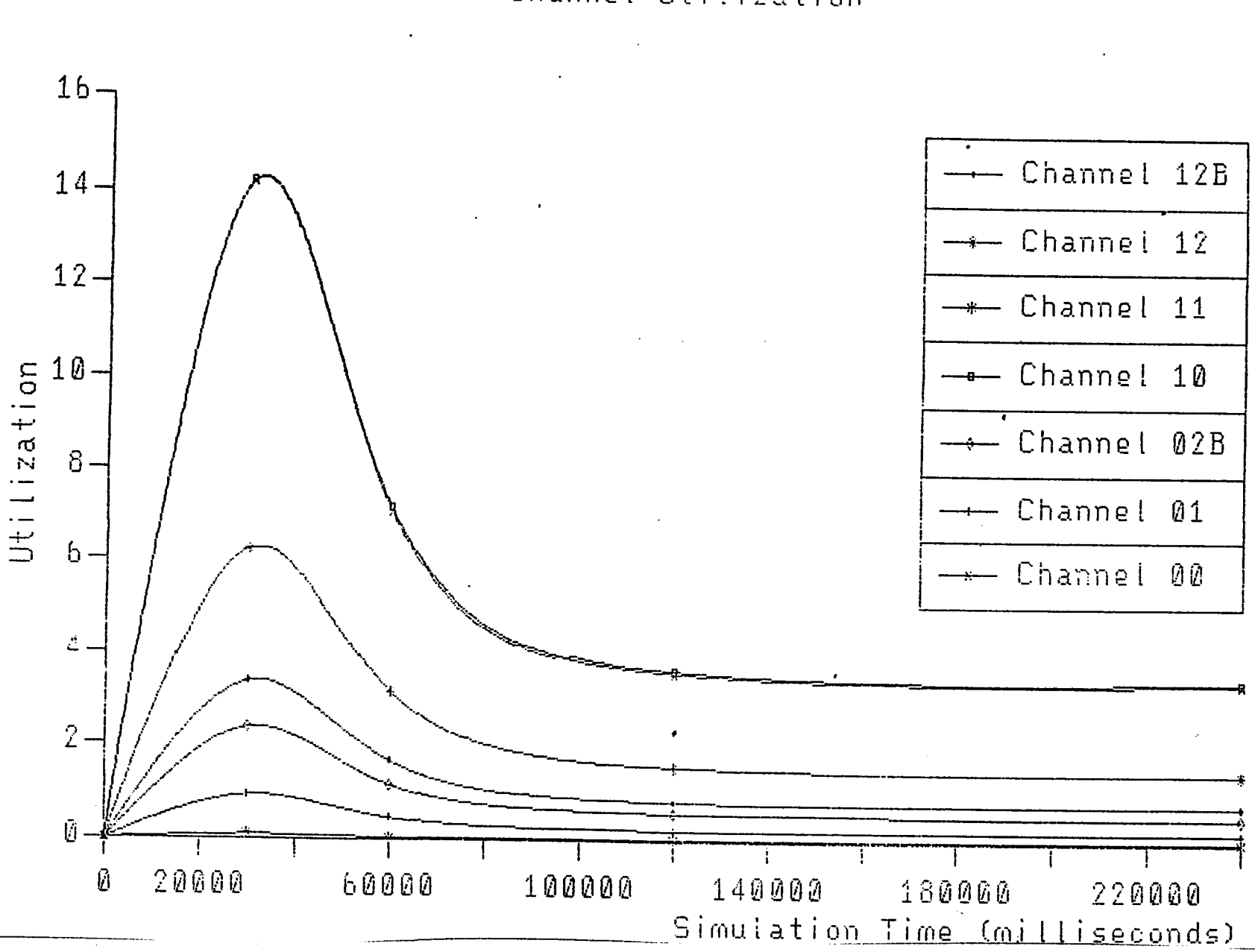


Figure 6.36 Effect of simulation time on Channel Utilization

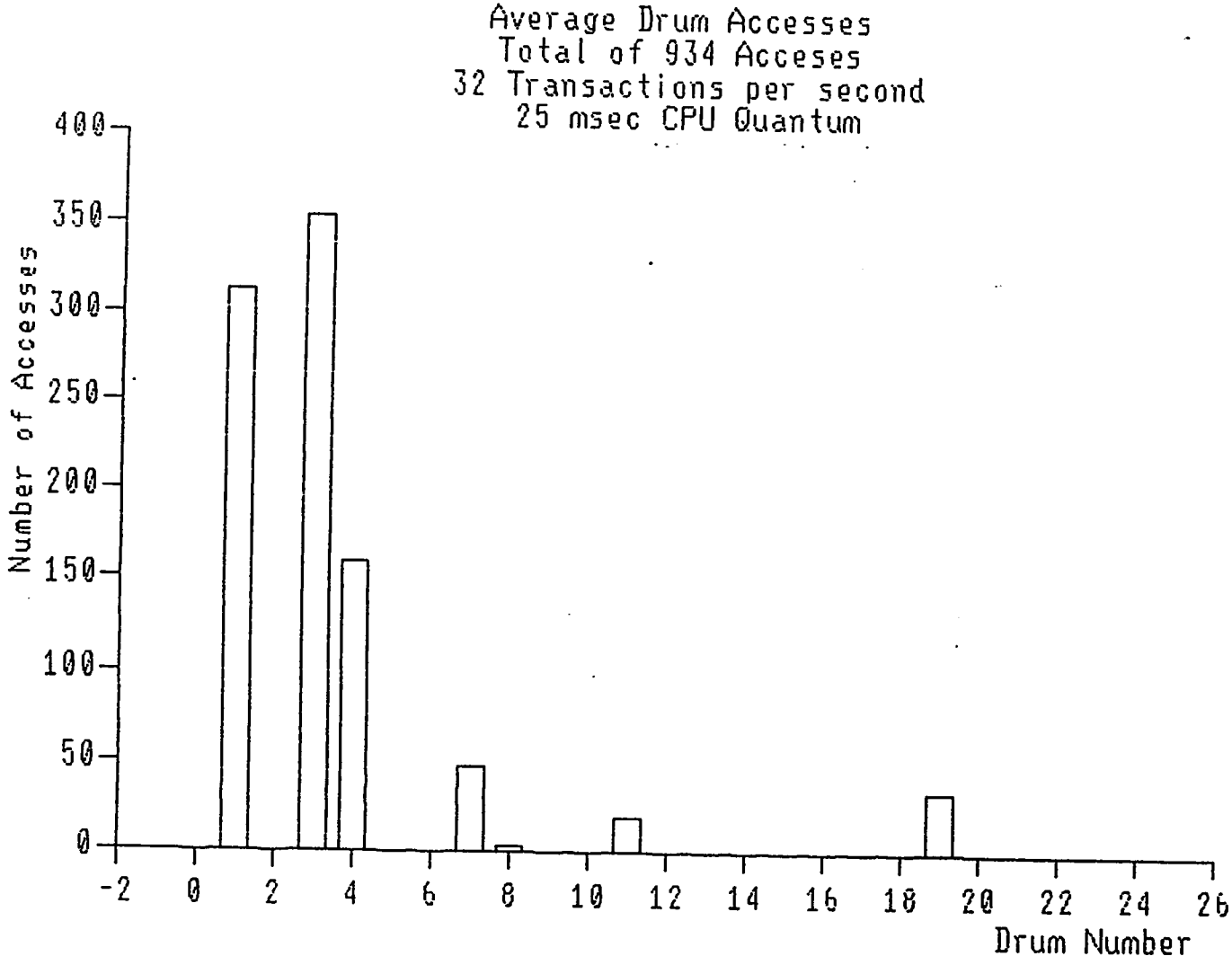


Figure 6.37 Average Number of Drum Accesses.

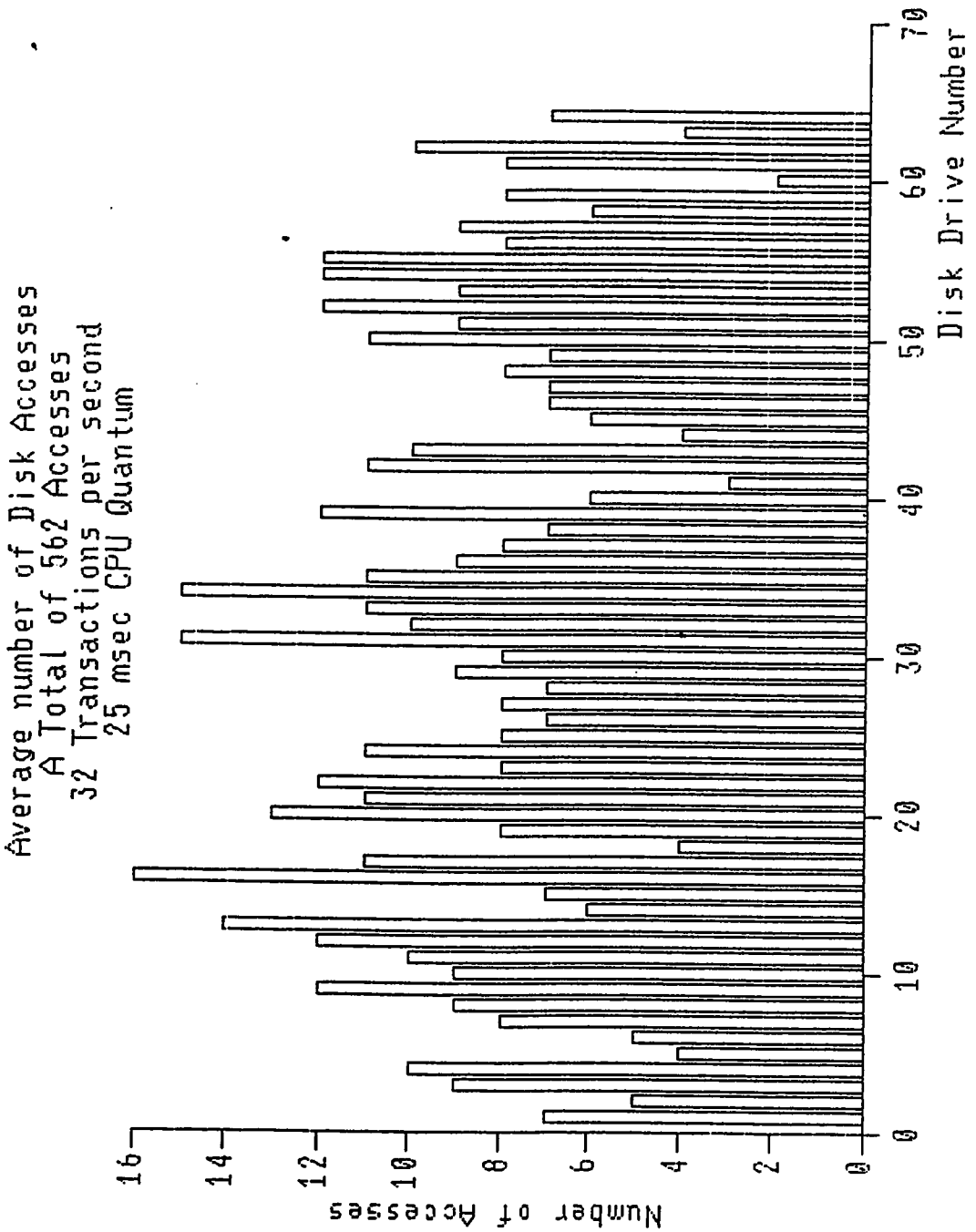
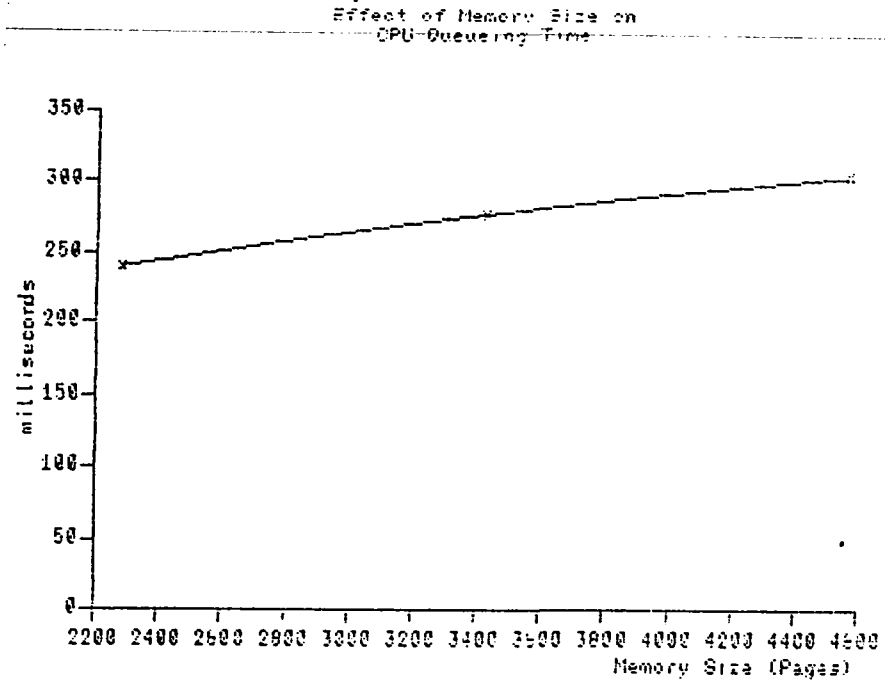
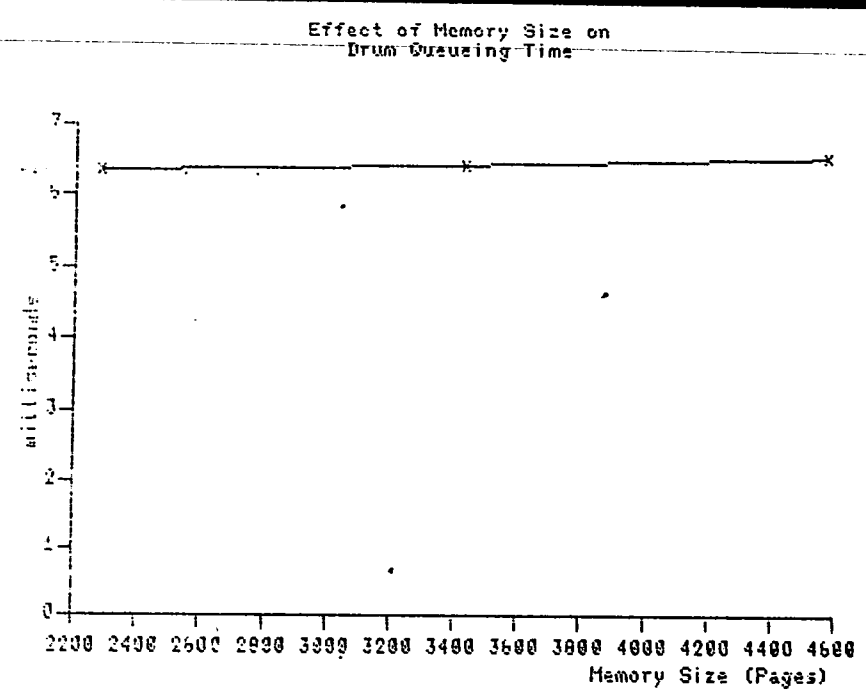
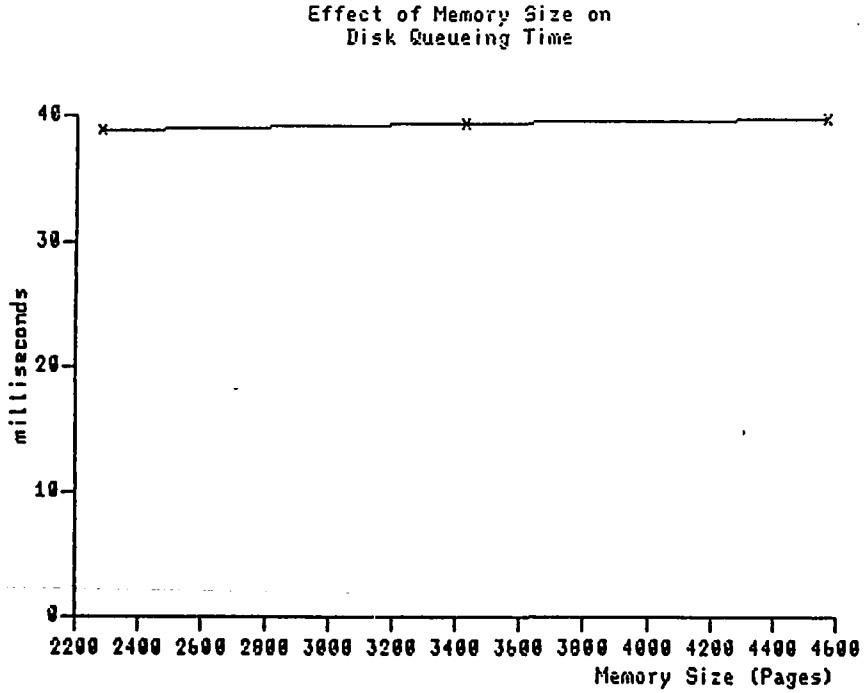


Figure 6.38 Average Number of Disk Accesses.







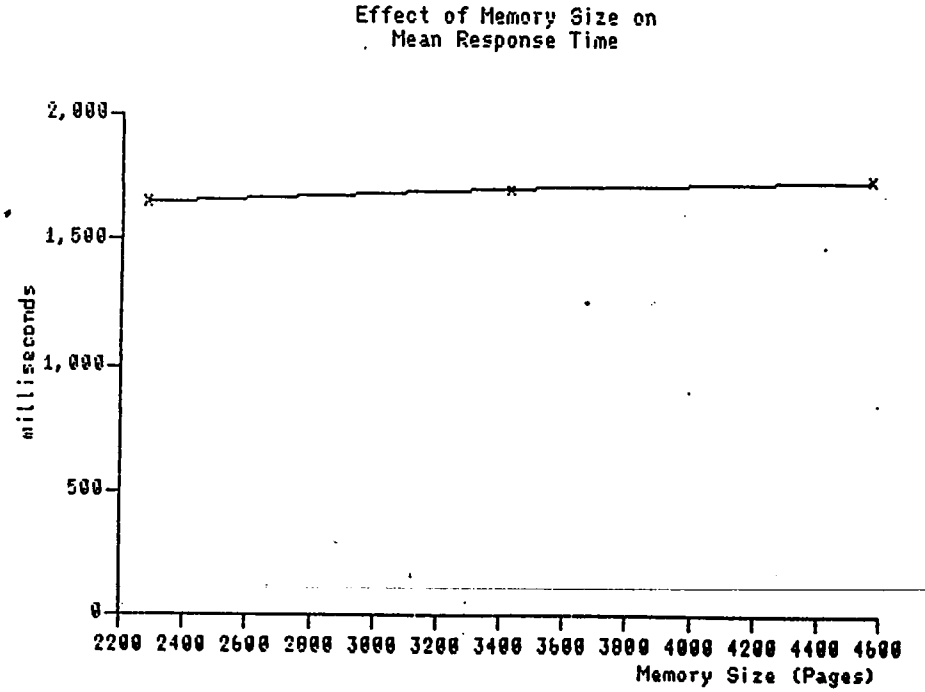
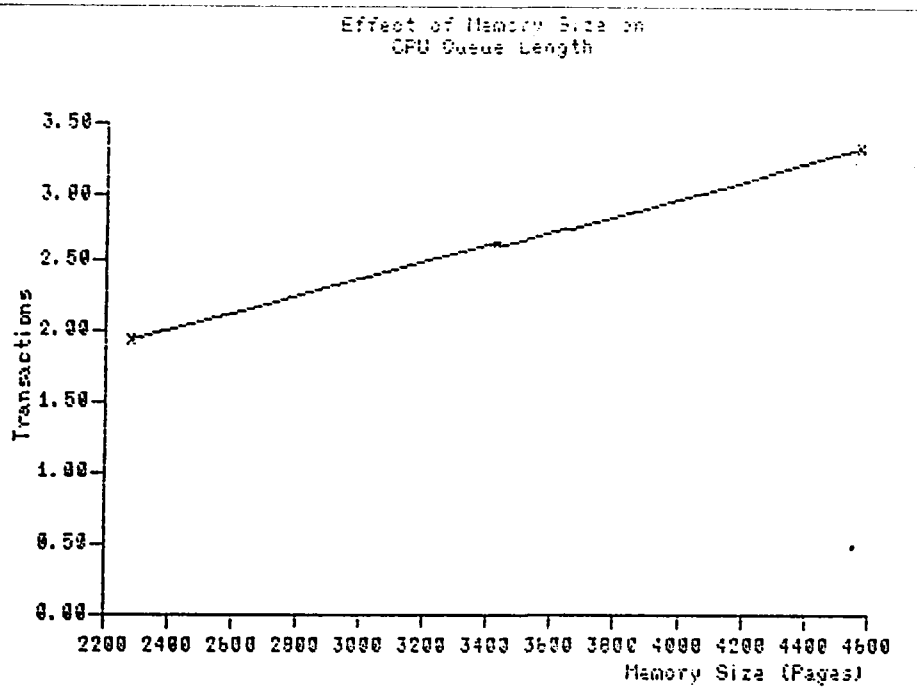
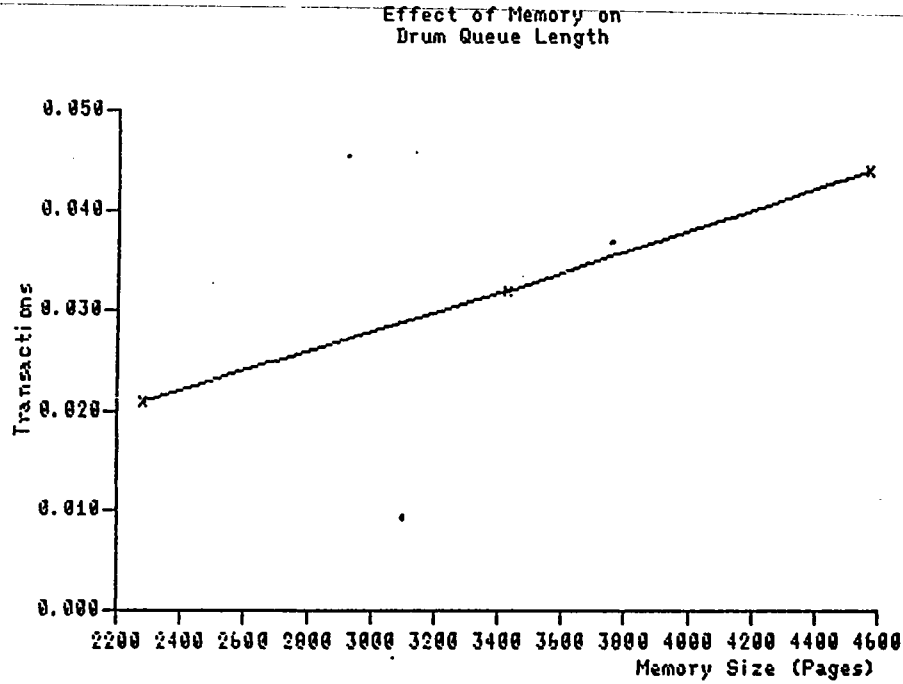
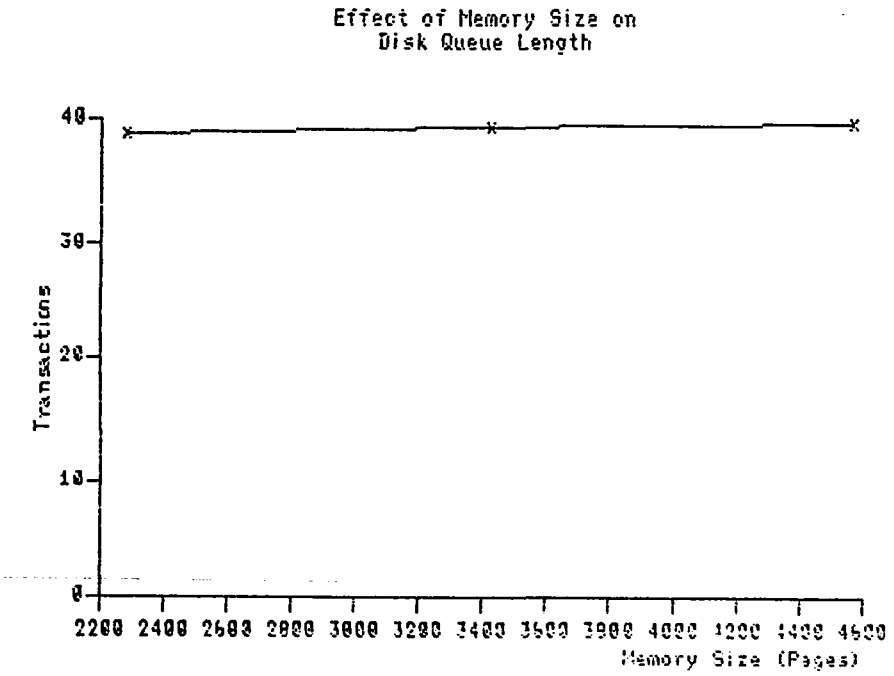


Figure 6.39 Effect of Memory Size on Queueing Time.







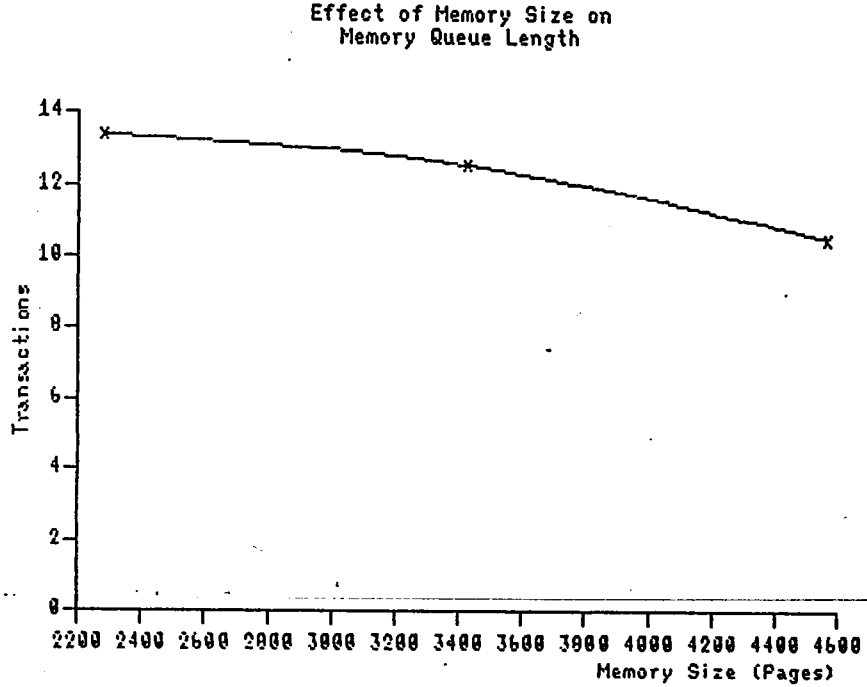
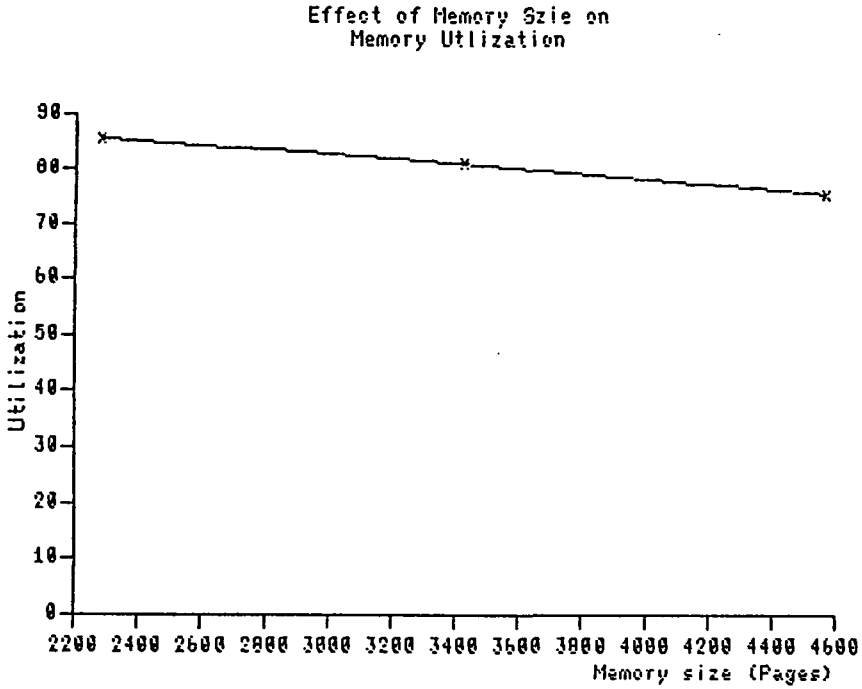
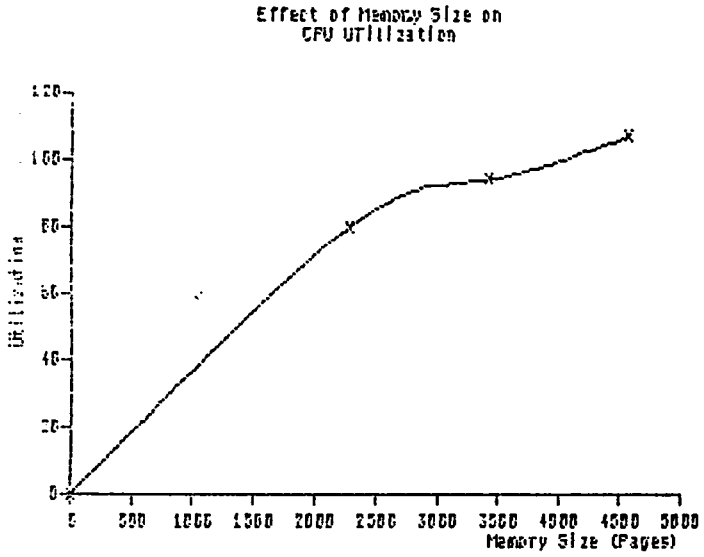
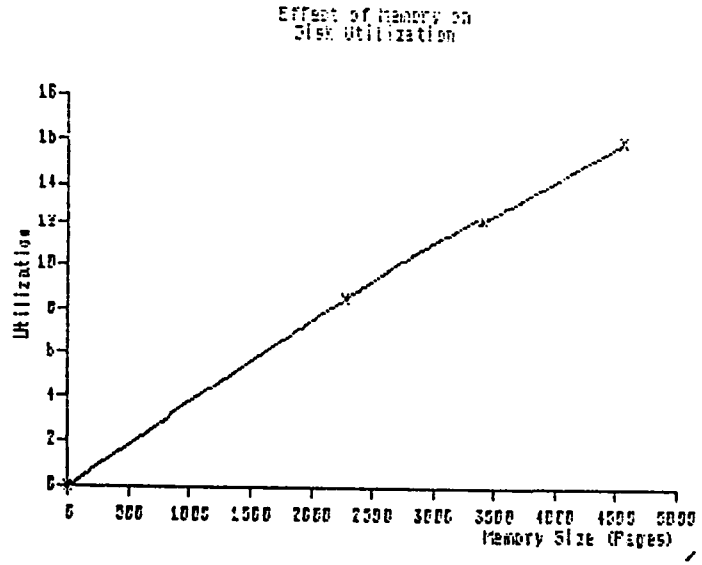


Figure 6.40 Effect of Memory Size on Queue Length.







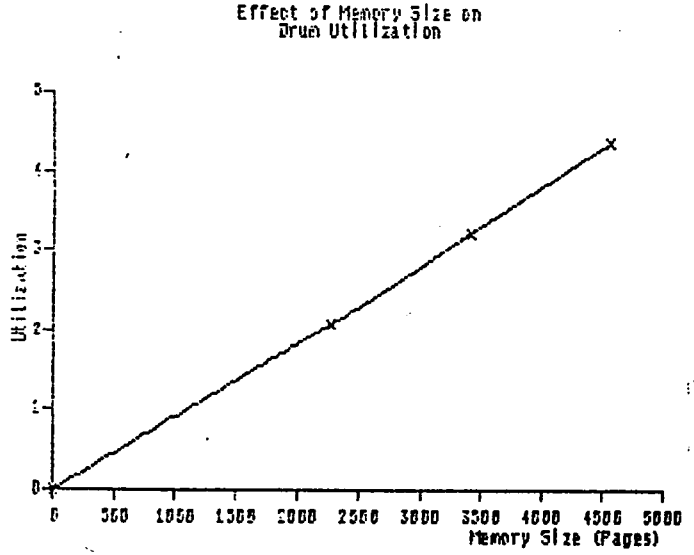


Figure 6.41 Effect of Memory Size on Utilization.

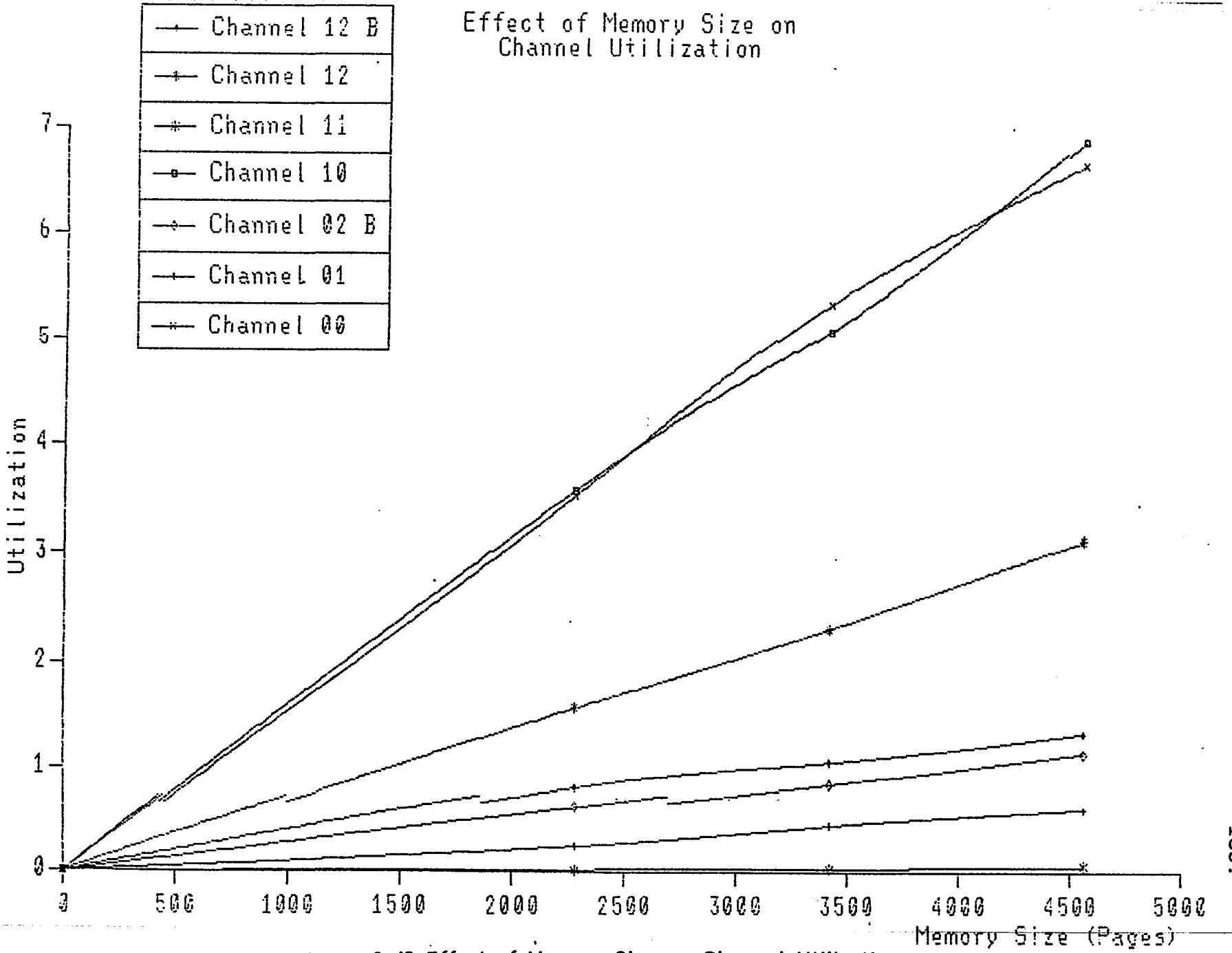


Figure 6.43 Effect of Memory Size on Channel Utilization.

7. CONCLUSION

The objective of this thesis was to develop a modeling methodology that would enable the construction of a model for the SAMIS Online System, for the purpose of studying the performance of that system.

The modeling methodology has been developed. A core model representing the basic SAMIS Online System configuration has been constructed. The modeling methodology developed to incorporate the Application Subsystems has been applied and tested in the core model on the Civil Registration Subsystem of the SAMIS Online System.

Following the construction of the core model and the representation of the Civil Registration Subsystem in the model, the model has been translated into a computer processable form in the PAWS simulation language. The model in PAWS language has then been tested and debugged.

Different experiments have been conducted on the model by running the simulation programs several times with various setups. The bahaviour of the model was observed and documented by collecting appropriate statistics. The experimental data have been presented in graphical form to enable the modeler to interpret trends in the behaviour of the model. The resulting graphs have been analyzed and their general characteristics have been identified.

The work described in this thesis did not involve all subsystems of the SAMIS Online System and also does not reflect the recent modifications in the hardware and software components of the modeled system, that came about after the project was launched. Additional work need to be carried out to make the existing model most up to date and fully representative of the operational aspects of the SAMIS Online System. Such additional work is summarized below.

1. The incorporation of the other SAMIS Online System Application Subsystems, such as the drivers licences, vehicle registration, pilgrim services, passport issuance, miscreant control, criminal records, alien control, border control, micrographics, and the communications subsystem. The modeling methodology described and demonstrated in this thesis need be applied to every remaining subsystem to create model initialization data about subsystem functions, task structures of functions, and file access properties of remaining tasks.

2. The incorporation of new updates on the evolving SAMIS Online System.

3. The refinement of parameters and assumptions made throughout the model construction phases, such as the I and D bank memory requirements of tasks.

4. The development of an interface that would

interactively display the queue formation on different servers as time elapses using graphics. This would enable the discovery of different bottlenecks forming in the model in continuouse time. In the current modeling methodology the model's behaviour shows the model's status at discrete intervals of time. This does not enable the detection of problems occuring in between the test intervals.

REFERENCES

1. Sauer, C. H., and K. M. Chandy. Computer Systems Performance Modeling. Printice-Hall Inc., Englewood Cliffs, N.J. 1981.

2. Ferrari, Dominico. Computer Systems Performance Evaluation. Prentice-Hall Inc., Englewood Cliffs, N.J., 1978.

3. Lazowska, E. D., et al. Quantitative System Performance.Computer System Analysis Using Queueing Network Models. Prentice-Hall Inc., Englewood Cliffs, N.J., 1984.

4. Ferrari, D., and S. Serazzi, A. Zeigner. Measurment and Tuning of Computer Systems. Prentice-Hall Inc., Englewood Cliffs, N.J., 1983.

5. Morris, M. F., and P. F. Roth. Tools and techniques, Computer Performance Evaluation for Effective Analysis. Van Nostrand Reinhold Company N.Y., 1982.

6. Zeigler, B. P. Theory of Modelling and Simulation. John Wiley & Sons Inc., 1976.

7. Borovits, I., and S. Neumann. Computer Systems Performance Evaluation. D.C. Heath and Company, 1979.

8. Greenberg, S. GPSS Primer. Wiley-Interscience, John Wiley & Sons Inc., 1972.

9. Barnes, M. F. Measurment and Modelling Methods for Computer Systems Performance Studies. Langton Information Systems Ltd, 1979.

10. PAWS 2.0, Performance Analyst Workbench System. User's Manual. Information Research Associates, Austin, Texas, 1984.

11. PAWS 2.0. An Introduction to PAWS. IRA, Austin, Texas,1984.

12. PAWS 2.0. Introduction and Technical Summary. IRA, Austin, Texas, 1983.

13. PAWS 2.0. Installation Guide. IRA, Austin, Texas, 1985.

14. Saudi Arabian Ministry of Interior System Documentation. Final Detailed Design Document for the Online System Vol 1 : Software System Specification,

C 45 FDD vol 1 C2 (Oct 1983)

15. Saudi Arabian Ministry of Interior System documentation.
Final Detailed Design Document for the Online System
Vol 2 : Data Base Specification (Oct 1983)

16. Saudi Arabian Ministry of Interior System documentation.
Functional Design Specification
Vol 1, Part 1 : Hardware System Specification
C 65 FDS vol 1 Part 1 C2 (Jan 1980)

17. Saudi Arabian Ministry of Interior System documentation.
Functional Design Specification
Vol 4 : Preliminary Data Base Specification,
C 65 FDS vol 4 C2 (Jan 1980)

18. Saudi Arabian Ministry of Interior System documentation.
Application Programmers Manual for the Online System
Vol 1 : System Design, Part 4 : Alien Control Element
C 85 PGM vol 1 Part 4 (Dec 1983)

19. Saudi Arabian Ministry of Interior System documentation.
Application Programmers Manual for the Online System
Vol 1 : System Design, Part 5 : Border Control Element
C 85 PGM vol 1 Part 5 (Dec 1983)

20. Saudi Arabian Ministry of Interior System documentation.
Application Programmers Manual for the Online System
Vol 1 : System Design, Part 8 : Pilgrim Control Element
C 85 PGM vol 1 Part 8 (Dec 1983)

21. Saudi Arabian Ministry of Interior System documentation.
Application Programmers Manual for the Online System
Vol 1 : System Design,
Part 9 : Civil Registration Element
C 85 PGM vol 1 Part 9 (Dec 1983)

22. Saudi Arabian Ministry of Interior System documentation.
Data Management Plan,
Vol 2 : Data Base Management Manual
C95 DMP vol 2 (Feb 1983)

23. Saudi Arabian Ministry of Interior System documentation.
MANAGE Data-Base Designer's Reference,
QB 301 MAN 253 C2 (Dec 1982)

24. Saudi Arabian Ministry of Interior System documentation.
MANAGE Procedural Language Reference,
QB 321 PRC 364 C2 (Aug 1982)

25. Saudi Arabian Ministry of Interior System documentation.
IMP (Interaction Management Protocol),
QB 311 IMP 29 C2 (April 1983)

26. Saudi Arabian Ministry of Interior System documentation.
3690 Communications Processor, M 84 SEC 13 C2 (1979)

27. Saudi Arabian Ministry of Interior System documentation.
Architecture Programmer Manual,
Presentation Layer System
Specification, Volume 1 : Data Structures,
A 61 ARC 26
Vol 1 (Sep 1983)

28. Saudi Arabian Ministry of Interior System documentation.
Presentation Layer Users Guide,
A 51 PRE 1982 C1 (Oct 1982)

29. Saudi Arabian Ministry of Interior System documentation.
CSTS II System Services Manual

30. Saudi Arabian Ministry of Interior System documentation.
Programmers Manual Online System
Volume 1 : Part 9 : Civil Registration.
C 91 .PGM Vol 1 Part 9 C1.

31. Trip Report by NP:SAMIS SANCST project team, task II.
Trip : May 1, 1985.
Report date : July 28, 1985.

32. Trip Report by NP:SAMIS SANCST project team, task II.
Trip : July 6, 1985.
Report date : July 29, 1985.

33. Trip Report by NP:SAMIS SANCST project team, task II.
Trip : July 16, 1985.
Report date : July 30, 1985.

34. Trip Report by NP:SAMIS SANCST project team, task II.
Trip : August 4, 1985.
Report date : August 17, 1985.

Appendix (A)

DESCRIPTION AND PROPERTIES OF SAMIS ONLINE SYSTEM FILES

File "FILES.DAT" has been extracted from ref [7] . The basic contents of the file are the the file name, maximum Record Size, file type, file volume, drum number on which the index is stored, and keysize.

Files could be Unkeyed, Hashed, Indexed Sequential, Unpunctuated, Punctuated, Library index, Relocatable object code, Executable absolute code.

This appendix will contain information on unkeyed, hashed, and indexed sequential files only.

```
* File      : FILES2.SCR
* Updated   : 15 October  1986
*             30 November 1986
* Done By   : Khaled W. Al-Dhaher
*
* Contents : This file contains the information about
* all the files in the SAMIS system, accessed by the
* various functions.
*
* ONLINE SYSTEM RECORD INFORMATION.
* ---------------------------------------------------
* NAME        Max Record ORG.  VOLUME(1990)  Key Size
*             Size in
*             Word,Byte                      Word,Type
* ---------------------------------------------------
* DRUM NUMBER IN COL 43 IS RANDOM
* KEY SIZE    IN COL 46 IS IN WORDS
*0000000011111111112222222222333333333344444444445555555556
*2345678901234567890123456789012345678901234567890123456890
*
ACCSTA (042)  95   380 INDEXED   100000   08 01 10 B
ALIEN  (036)  50   200 HASHED   3850000   15 01 10 B
ALNNBR (067)   4    16 HASHED         3   05 01  1 B
ALTRNS (080)   9    36 INDEXED    20000   09 04 16 G
ARREST (012)  91   364 INDEXED    95400   15 01 10 B
BDRNBR (071)   4    16 HASHED         2   02 01  1 B
CITZEN (016)  35   140 HASHED   6748950   07 01 10 B
CIVHOH (077)  29   116 HASHED    674895   19 01 10 B
CIVNBR (068)   4    16 HASHED         6   19 01  1 B
CMEVNT (001)  35   140 INDEXED   142800   01 01 10 B
CMHIST (008)   4    16 HASHED     95400   18 01  7 B
CMNAMX (066)  52   208 INDEXED    95400   16 01 10 B
CMPSPT (003)  53   212 HASHED    142800   14 01 10 B
CMSTAT (084)  16    64 UNKEYED        1   07 00  0 Unkeyed
CMVEHC (004)  69   276 HASEHD    142800   10 01 10 B
CRMNBR (070)   4    16 INDEXED      540   03 02  5 G
CVSTAT (069)  35   140 INDEXED      200   01 02  8 G
DEPDEP (065)  55   220 INDEXED     7500   09 02  6 G
DEPORT (011)  57   228 HASHED      3000   02 01  7 B
DLFEES (093)  51   204 INDEXED      484   11 02  8 G
DLSTAT (094)  59   236 INDEXED      273   09 02  8 G
DYSTAT (090)   9    36 INDEXED    10000   24 06 22 G
EXSVIO (038)  39   156 INDEXED   320000   15 02  8 G
FGRINX (013)   4    16 INDEXED    95400   11 05 20 C
FORDRI (037)  38   152 INDEXED    20000   04 01 10 B
LQUOTA (025)   5    20 INDEXED   600000   14 03 10 G
LSSTAT (091)   9    36 INDEXED    10000   12 06 22 G
MALIAS (058)  68   272 INDEXED    26667   14 02  8 G
MBADDR (48)    5    20 INDEXED     8800   17 02  8 G
MBGEN  (33)    4     6 UNKEYED        1   14 00  0 Unkeyed
MBLACK (046) 242   968 HASHED     60000   16 01  7 B
MBTEXT (051) 312 1,248 INDEXED     2900   14 01  6 B
MCHECK (059)  63   252 INDEXED     1000   08 02  6 G
MECREG (026)  27   108 INDEXED    15000   12 01 10 B
```

```
MICROA (092)   6    24 INDEXED 15489234   24 02  6 G
MICROF (045)   6    24 INDEXED  1296000   13 05 19 G
MICROI (043)   4    16 INDEXED     2000   09 02  8 G
MICROP (044)  10    40 INDEXED    13500   12 03 12 G
MIDGEN (088)   5    20 HASHED         2   13 01  1 B
MSENCT (047)  51   204 INDEXED    30000   01 03 12 G
MSSTAT (053)  40   160 UNKEYED        1   11 00  0 Unkeyed
MSTRSM (052)   7    28 INDEXED      650   10 01 10 B
MVFEES (084)  33   132 INDEXED      110   10 02  8 G
MVSTAT (035)  63   252 INDEXED      110   01 02  8 G
PGEADD (055)  76   304 HASHED      1000   15 02  8 G
PGENBR (054)   4    16 UNKEYED        1   24 00  0 Unkeyed
PGESTA (056)  15    60 INDEXED    16000   10 01  4 B
PGEXTD (057) 248   992 HASHED    150000   11 01 10 B
PILGRM (056)  39   156 HASHED   1000000   02 01 10 B
PRECHK (076)   9    36 HASHED      9000   07 02  8 G
PRIRES (039)  33   132 HASHED    390000   11 01 10 B
PRISON (010)  31   124 INDEXED   114000   15 01 10 B
PRIVAT (081)  39   156 INDEXED    57000   11 01 10 B
PSPLOG (041)  47   188 HASHED      1000   08 01  5 B
PSPORT (019)  62   248 INDEXED  1600000   14 02  6 G
SAMORD (022) 198   792 INDEXED   177180   11 01  8 B
SDDRLC (027)  28   112 HASHED   1500000   18 01 10 B
SERIAL (006)  55   220 HASHED    142800   21 01 10 B
SPONSR (023)  36   144 INDEXED   100000   06 01 10 B
TERMF  (...)   5    20 INDEXED     2000   10 01  4 C
TRAVEL (031)  14    56 HASHED   6056785   02 01 10 B
TRAVIO (032)  42   168 HASHED   1450000   03 02  8 G
TRVDEP (074)  11    44 INDEXED  1815834   15 02  8 G
UNSERL (005)  50   200 HASHED    142800   06 01 10 B
USERF  (063)   5    20 INDEXED     4000   21 02  6 C
VEHEXS (049)  33   132 INDEXED    30000   02 02  8 C
VEHOWN (050)  83   332 INDEXED   415000   23 02  8 C
VEHREG (028)  50   200 HASHED   1450000   03 02  8 C
VICTIM (002)  25   100 HASHED    142800   09 01 10 B
VIOBOK (029)   8    32 INDEXED    60000   05 02  8 G
VISA   (040)  10    40 HASHED   7700000   09 01  8 B
VISDEP (075) 112   448 INDEXED   251860   03 02  8 G
VISITR (060)  79   316 INDEXED   839540   09 01 10 B
* ------------------------------------------------
* VIRTUAL RECORDS: NOT INCORPORATED IN THE MODEL
* ------------------------------------------------
* Set : HOHGRP
*             Base Record       : CITZEN
*             Descendant Record : CIVHOH
*             Relation          : 1 to 1
* Set : VTRAVL
*             Base Record       : TRAVEL
*             Descendant Record : VISITR
*             Relation          : 1 to 1
* Set : SPNALN
*             Base Record       : ALIEN
*             Descendant Record : SPONSR
*             Relation          : 1 to N
```

```
* Str : VISARP
*              Base Record       : VISA
*              Descendant Record : ALIEN,SPONSR
*              Set : ALVISA
*                           Record   : VISA
*                           Record   : ALIEN
*                           Relation : 1 to N
*              Set : SPNALN
*                           Record   : SPONSR
*                           Record   : ALIEN
*                           Relation : 1 to N
* Set : QUOTA
*              Base Record       : SPONSR
*              Descendant Record : LQUOTA
*              Relation          : 1 to N
* Str : VISITRV
*              Base Record       : VISITR
*              Descendant Record : TRAVEL,VISDEP
*              Set : VTRAVL
*                           Record   : VISITR
*                           Record   : TRAVEL
*                           Relation : 1 to 1
*              Set : VDEP
*                           Record   : VISITR
*                           Record   : VISDEP
*                           Relation : 1 to N
* Str : TRVSTT
*              Base Record       : TRAVEL
*              Descendant Record : CITZEN,ALIEN,VISITR
*              Set : STRAVL
*                           Record   : TRAVEL
*                           Record   : CITZEN
*                           Relation : N to 1
*              Set : ATRAVL
*                           Record   : TRAVEL
*                           Record   : ALIEN
*                           Relation : N to 1
*              Set : ??????
*                           Record   : TRAVEL
*                           Record   : VISITR
*                           Relation : N to 1
* Set : HOHGRP
*              Base Record       : CIVHOH
*              Descendant Record : CITZEN
*              Relation          : 1 to 1
* Set : MCXPG
*              Base Record       : MICROP
*              Descendant Record : MICROI
*              Relation          : 1 to N
* ----------------------------------------
* Online System Auxiliary Files
* ----------------------------------------
* 01-SAMIS Support Files
* ----------------------------------------
```

```
MISNMX (087)  39   156 INDEXED   180000   13 16 64 G
NAMEX  (085) 151   604 INDEXED  8000000   04 16 64 G
* 14,090,303 Accesses/Year
* 11,271,236 are for update
UNKNMX (086)  39   156 INDEXED      500   11 16 64 G
* ----------------------------------------
* 02-SAMIS Archive Files
* THESE FILES ARE NOT ACCESSED BY THE ONLINE SYSTEM.
* THEY WILL NOT BE USED, THEY ARE LISTED FOR COMPLETENESS.
* ----------.-----------------------------
* ALNARC      50     200 TAPE     89375    1 10 B
* BCARCH      6516 26063 TAPE    574361    1  8 B
* DELIDX      17      68 INDEXED   10000    6 23 G
* PGEARC      44     176 TAPE      20000    1  4 B
* PGEARL                                   1 10 B
* PGOVDU                                   1 10 B
*
* ----------------------------------------
*
* UPDATE : 30 NOVEMBER 1986
*
* THE ACTIVITY FILE IS A SPECIAL FILE THAT WILL BE
* CONSIDERED AS ANY INDEXED FILE, ALTHOUGH IT IS NOT.
* THE IDEA, IS THAT THIS FILE WILL INITIATE ONE DRUM
* I/O FOLLOWED BY O  NE DISK I/O. TO ACHIEVE THIS IT
* IS ASSUMED IT IS AN INDEXED FILE, WITH THE DESCRIBED
* INFORMATION, WHICH WILL INITIATE THE REQUIRED
* DRUM,DISK I/O'S.
*
AF     (...)  01   001 INDEXED      001   03 01 01
$ TO INDICATE THE END OF THE FILE.
```

Appendix (B)

TASK STRUCTURE OF SAMIS ONLINE SYSTEM FUNCTIONS.

A method of representing the probabilistic sequence of task executions given a SAMIS ONLINE System function has been developed. The method has currently been applied to represent the task structure of all the functions of Civil Registration Element.

An examination of SAMIS ONLINE System functions show that the sequence of tasks executed is not deterministic and can be data-dependent. This means that the stepcode of the task that is to be executed next depends on data entered by an operator on a terminal, such as menu selections. This fact has been modeled by using probabilistic sequences of task stepcodes. If a task is followed by n possible tasks and if there is no additional information on the frequency of execution of the successor tasks, a probability of (1/n) is assigned for the execution of any successor task. That is, each successor task is equally likely to be executed.

The task structure for all functions of the Civil Registry Element can be seen in the rest of this appendix. The following is a brief summary of the format that has been used to represent the task structure of functions.

1. The first field is the stepcode of the first task to be executed for the designated function.

2. The second field contains one hundred times the probability of control passing to other stepcodes.

3. The third field contains the stepcode of the next task to which control of execution will be passed with the probability given in the second field.

4. A blank in the first field implies that there are more than one successor tasks to the task which appeared in last field one.

5. A (-1) at column one, means the end of the description of the tasks for the current function.

6. A second (-1) at the end appears to indicate the end of functions in the current element.

The file listing follows.

```
*-------------------------------------------------
* FILE           : STEPCD.DAT
* DONE BY        : KHALED W. AL-DHAHER
* CREATED        :  1 OCTOBER  1986.
* MAJOR UPDATES : 15 OCTOBER  1986,
*                  27 NOVEMBER 1986.
*
* CONTENTS : INFORMATION ABOUT THE TASK STRUCTURE OF THE
* SAMIS ONLINE SYSTEM FUNCTIONS.
* IT DESCRIBES THE FLOW OF CONTROL FROM ONE "STEPCODE",
* AS EACH SECTION OF THE TASK IS CALLED, TO THE OTHER
* STEPCODES. THERE MIGHT BE MORE THAN ONE BRANCH, TO
* DIFFERENT STEPCODE, A PROBABILITY IS DESIGNATED TO EACH
* BRANCHING ARC  TO IDENTIFY THE POSSIBILITY OF CONTROL
* PASSING TO IT. AT THE END OF DESCRIPTION OF EACH FUNCTION,
* A -1 WILL  BE ENCOUNTERED. AT THE END OF DESCRIPTION OF
* EACH ELEMENT   A -1 WILL AGAIN BE ANCOUNTERED.
*
*-------------------------------------------------
* 09  :CIVIL REGISTRATION ELEMENT
*-------------------------------------------------
0901 : HEAD OF HOUSEHOLD CITIZENSHIP REGAINING
*-------------------------------------------------
9021   50 9022 (MENUE) FOR HOH CITIZENSHIP REGAINING
       50 9006 (MENUE) FOR DEPENDANT CITIZINSHIP REGAINING
9022   25 9003
       25 9023
       25 9024
       25 9025
9003  100 9004
9024   50 9003
       50 9023 * only if it first passes through 9023.
9025   50 9003
       50 9023 * only if it first passes through 9023.
9023   25 9023 Multiple screen display.
       25 9003
       25 9024
       25 9025
*-------------------------------------------------
* 0901 : DEPENDANT CITIZENSHIP REGAINING
*-------------------------------------------------
9006  100 9007
9007  100 9004
9004   10 9004 Multiple screen display.
       15 9027
       15 9028
       15 9029
       15 9030
       15 9031
       15 9032
9027   50 9027 Multiple screen display.
       50 9032
9028   50 9028 Multiple screen display.
       50 9032
```

```
9029  50 9029 Multiple screen display.
      50 9032
9030  50 9030 Multiple screen display.
      50 9032
9031  50 9031 Multiple screen display.
      50 9032
9032 100 9033
9033 000 0000 NULL
  -1
*-----------------------------------------------------
* THIS IS THE LAST STEPCODE IN THIS FUNCTION.
* THIS MEANS CONTROL NORMALLY SHOULD GO TO THE MENUE,
* BUT SCINCE WE WANT TO REPRESENT THE HANDLING OF (ONE)
* COMPLETE PASS IN THE STEPCODE LIST, REACHING HERE
* MEANS FINISHING THE TASK REQUIRED.
*-----------------------------------------------------
0902 : HEAD OF HOUSEHOLD NATURALIZATION
*-----------------------------------------------------
9074  50 9075 (MENUE) HOH NATURALIZATION
      50 9077 (MENUE) DEPENDANT NATURALIZATION
9075 100 9003
*-----------------------------------------------------
* 0902 : DEPENDANT NATURALIZATION
*-----------------------------------------------------
9077 100 9007
9007 100 9076
9075 100 9003
9003 100 9076 (* see 9003 to 9004 in function 0901)
9076 100 9004
9004 100 9027
9027  50 9027 Multiple screen display.
      50 9028
9028  50 9028 Multiple screen display.
      50 9029
9029  50 9029 Multiple screen display.
      50 9030
9030  50 9030 Multiple screen display.
      50 9031
9031  50 9031 Multiple screen display.
      50 9083
9083 100 9084
9084 000 0000 NULL
  -1
*-----------------------------------------------------
0903 : REGISTER HEAD OF HOUSEHOLD
*-----------------------------------------------------
9001  50 9085 (MENUE) REGISTER HOH
      50 9120 (MENUE) REGISTER DEPENDANT
9085 100 9086
*-----------------------------------------------------
* 0903 : REGISTER DEPENDENTS
*-----------------------------------------------------
9120 100 9121
9121 100 9008
```

```
9008 100 9009
9009 100 9086 ADDED JUST TO MAKE 9086 NULL FOR BOTH.
9086 000 0000 NULL
  -1
*-----------------------------------------------------
0904 : ISSUE NATIONAL IDENTIFICATION CARD
*-----------------------------------------------------
9001  50 9010 (MENUE) ISSUE  NATIONAL ID
      50 9013 (MENUE) ENCODE NATIONAL ID
9010 100 9011
9011 100 9012
*-----------------------------------------------------
* 0904 : ENCODE NATIONAL IDENTIFICATION CARD
*-----------------------------------------------------
9013 100 9014
9014 100 9015
9015 100 9005
*9005 000 0000 NULL
9005 100 9012 ADDED TO MAKE 9012 A COMMON NULL
9012 000 0000 NULL
  -1
*-----------------------------------------------------
0905 : FAMILY BOOK
*-----------------------------------------------------
9001 100 9036 (MENUE)
9036  50 9037 APPLICANT IS DEPENDANT (CHECK PROB.)
      50 9038 APPLICANT IS HEAD OF HOUSEHOLD
9037 100 9038
9038 100 9009
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0906 : REGISTER MARRIAGE
*-----------------------------------------------------
9001 100 9040 (MENUE)
9040  50 9041 (HUS & OR WIFE ARE DEPENDANTS)
      50 9042 (HUS & WIFE ARE HEADS OF HOUSEHOLDS)
9041  33 9042 (HUS-HOH)
      33 9043 (WIF-HOH)
      33 9044 (WIF-DEP)
      33 9045 (HUS-DEP)
9042  50 9043 (WIF-HOH)
      50 9044 (WIF-DEP)
9045  50 9043 (WIF-HOH)
      50 9044 (WIF-DEP)
9043 100 9009
9044 100 9009
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0907 : REGISTER DIVORCEE
*-----------------------------------------------------
9001 100 9046 (MENUE)
9046 100 9047
```

```
9047  20 9048 (WIF-HOH) Very rare to find a wife as a HOH
      80 9049 (WIF-DEP) So let's assume a 1:5 prob.1/5=0.2
9048 100 9122
9049  50 9050
      50 9051
9050 100 9122
9051 100 9124
9122 100 9123
9123 100 9009
9124 100 9009
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0908 : REGISTER DEATH
*-----------------------------------------------------
9001 100 9052 (MENUE)
9052  17 9053 (HOH) assume 1 family : 1 HOH & 4 DEP
      66 9054 (DEP) prob. that he is a dep is 4:1 + 1 for
      17 9009 (MICROGRAPHICS) 6prob 4*(1/6)=0.66
9053 100 9052
9054 100 9052
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0909 : WITHDRAW NATIONALITY
*-----------------------------------------------------
9074 100 9061 (MENUE)
9061  50 9062 (ID# ENTERED)
      50 9063 (NAME&DOB ENTERED)
9062 100 9063
9063 000 0000 NULL
  -1
*-----------------------------------------------------
0910 : TRANSFER REGISTRATION
*-----------------------------------------------------
9001 100 9055 (MENUE)
9055 100 9056
9056 100 9057
9057 100 9009
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0911 : UPDATE
*-----------------------------------------------------
9001 100 9058 (MENUE)
9058  33 9059
      33 9060
      33 9088
9059 100 9009
9060  50 9009
      50 9117
9088  50 9009
      50 9118 9018 IN NEW MANUAL
9118 100 9009
```

```
9117 100 9009
9009 000 0000 NULL
  -1
*-----------------------------------------------------
0912 : RECORD CHANGE DATE OF BIRTH
*-----------------------------------------------------
9002 100 9064 (MENUE)
9064  50 9065 (HOH)
      50 9066 (DEP)
9065 100 9068
9066 100 9068
9068 100 9070
9070 000 0000 NULL
  -1
*-----------------------------------------------------
0914 : QUERY
*-----------------------------------------------------
9001 100 9035 (MENUE)
9002 100 9035 (MENUE)
9074 100 9035 (MENUE)
9021 100 9035 (MENUE)
*     QUERY BY NATIONAL IDENTIFICATION NUMBER
9035  10 9089 (ACTIVITY 1)
      10 9092 (ACTIVITY 2)
*     QUERY BY ALIEN IDENTIFICATION NUMBER
      10 9095 (ACTIVITY 3)
      10 9097 (ACTIVITY 4)
*     QUERY BY INDIVIDUAL'S NAME
      10 9099
*     QUERY BY SAMIS ORDER NUMBER
       5 9106
       5 9107
       5 9108
       5 9109
       5 9110
*     SAMIS ORDER QUERY BY IND.NAME OR BY NATIONAL ID
       5 9112
       5 9113
       5 9114
       5 9115
       5 9116
*
9092 100 9093
9093 100 9999 NULL
9089  33 9090
      33 9091
      33 9999 NULL
9090  33 9090 MULTIPLE SCREEN DISPLAY
      33 9091
      33 9999 NULL
9091  50 9091 MULTIPLE SCREEN DISPLAY
      50 9999 NULL
*-----------------------------------------------------
* 0914 : QUERY BY ALIEN IDENTIFICATION NUMBER
```

```
*-----------------------------------------------------
9095  50 9096
      50 9999 NULL
9096  50 9096 MULTIPLE SCREEN DISPLAY
      50 9999 NULL
9097 100 9999 NULL
*-----------------------------------------------------
* 0914 : QUERY BY INDIVIDUAL'S NAME
*-----------------------------------------------------
9099  50 9100 High prob. because there is more displays
      25 9103
      25 9104
9103 100 9999 NULL
9104 100 9999 NULL
9100  33 9101
      33 9102
      33 9999 NULL
9101 100 9100
9102 100 9100
*-----------------------------------------------------
* 0914 : QUERY BY SAMIS ORDER NUMBER
*-----------------------------------------------------
9106 100 9999 NULL
9107 100 9999 NULL
9108 100 9999 NULL
9109 100 9999 NULL
9110 100 9999 NULL
*-----------------------------------------------------
* 0914 : SAMIS ORDER QUERY BY IND.NAME OR BY NATIONAL ID
*-----------------------------------------------------
9112  50 9113
      50 9999 NULL
9113  50 9114
      50 9999 NULL
9114  50 9115
      50 9999 NULL
9115  50 9116
      50 9999 NULL
9116 100 9999 NULL
*-----------------------------------------------------
* 0914 : MENUE PROCESSING
*-----------------------------------------------------
*
9999 000 0000 SINK FOR ALL STEPCODES IN FUNCTION 14.
*
  -1
  -1
```

Appendix (C)

SAMIS ONLINE SYSTEM TASKS AND THEIR PROPERTIES.

The information regarding the properties of all tasks that are used in the SAMIS ONLINE System have been placed into a file. This file is to be processed by the usernode to construct a table of task information.

The first field in this file indicates the module number that uniquely identifies the module. The stepcode associated with the module also uniquely identifies it, and is listed in the second field.

The third field lists the I-bank memory requirements for the task associated with the module listed in the first field, and the fourth field lists the task's D-bank memory requirements.

The fifth field lists whether the task is created at system startup or not. This is used as a flag wich is set to '0' to indicate task autocreation, and '1' to indicate otherwise.

The sixth field lists the number of waiting terminals after which another task copy could be created to enhance system performance.

The seventh field lists the maximum number of allowable task copies that could be created during system operation.

The eighth and ninth fields show the task kill time. Kill time number one in the eigth field is for original tasks, and Kill time number two in the ninth field is for task copies. A task will be sunk if it is inactive for a duration more than the indicated kill time.

This appendix contains a listing of the data provided by the abstraction process that represent details of the SAMIS Online System tasks and their properties.

```
CR0115 9022 00027 00010 0 001 05 1440 0001
CR0116 9023 00018 00006 0 001 05 1440 0001
CR0117 9024 00004 00002 0 001 05 1440 0001
CR0118 9025 00004 00002 0 001 05 1440 0001
CR0119 9003 00023 00008 0 001 05 1440 0001
CR0122 9004 00037 00013 0 001 05 1440 0001
CR0120 9006 00019 00007 0 001 05 1440 0001
CR0121 9007 00007 00002 0 001 05 1440 0001
CR0123 9034 00004 00002 0 001 05 1440 0001
CR0124 9026 00004 00002 0 001 05 1440 0001
CR0125 9027 00004 00002 0 001 05 1440 0001
CR0126 9028 00004 00002 0 001 05 1440 0001
CR0127 9029 00004 00002 0 001 05 1440 0001
CR0128 9030 00004 00002 0 001 05 1440 0001
CR0129 9031 00004 00002 0 001 05 1440 0001
CR0130 9032 00030 00011 0 001 05 1440 0001
CR0131 9033 00004 00002 0 001 05 1440 0001
CR0201 9075 00014 00005 0 001 05 1440 0001
CR0203 9077 00019 00007 0 001 05 1440 0001
CR0205 9076 00015 00005 0 001 05 1440 0001
CR0207 9083 00033 00012 0 001 05 1440 0001
CR0214 9084 00004 00002 0 001 05 1440 0001
CR0301 9085 00023 00008 0 001 05 1440 0001
CR0302 9086 00043 00016 0 001 05 1440 0001
CR0312 9087 00004 00002 0 001 05 1440 0001
CR0304 9120 00018 00007 0 001 05 1440 0001
CR0306 9121 00007 00002 0 001 05 1440 0001
CR0308 9008 00044 00016 0 001 05 1440 0001
CR0310 9009 00004 00002 0 001 05 1440 0001
CR0430 9010 00018 00007 0 001 05 1440 0001
CR0432 9011 00021 00008 0 001 05 1440 0001
CR0434 9012 00004 00002 0 001 05 1440 0001
CR0439 9013 00017 00006 0 001 05 1440 0001
CR0441 9014 00005 00002 0 001 05 1440 0001
CR0443 9015 00005 00002 0 001 05 1440 0001
CR0445 9005 00004 00002 0 001 05 1440 0001
CR0501 9036 00021 00007 0 001 05 1440 0001
CR0503 9037 00015 00006 0 001 05 1440 0001
CR0505 9038 00025 00009 0 001 05 1440 0001
CR0601 9040 00023 00008 0 001 05 1440 0001
CR0603 9041 00004 00002 0 001 05 1440 0001
CR0605 9042 00007 00002 0 001 05 1440 0001
CR0607 9043 00024 00009 0 001 05 1440 0001
CR0609 9044 00023 00009 0 001 05 1440 0001
CR0611 9045 00007 00002 0 001 05 1440 0001
CR0701 9046 00019 00007 0 001 05 1440 0001
CR0702 9047 00018 00006 0 001 05 1440 0001
CR0703 9048 00007 00002 0 001 05 1440 0001
CR0704 9049 00017 00006 0 001 05 1440 0001
CR0705 9050 00007 00002 0 001 05 1440 0001
CR0706 9051 00017 00006 0 001 05 1440 0001
CR0707 9122 00013 00005 0 001 05 1440 0001
CR0708 9123 00025 00009 0 001 05 1440 0001
CR0709 9124 00024 00009 0 001 05 1440 0001
```

```
CR0801 9052 00018 00006 0 001 05 1440 0001
CR0803 9053 00017 00006 0 001 05 1440 0001
CR0805 9054 00015 00006 0 001 05 1440 0001
CR0901 9061 00026 00009 0 001 05 1440 0001
CR0903 9062 00023 00008 0 001 05 1440 0001
CR0905 9063 00004 00002 0 001 05 1440 0001
CR1001 9055 00015 00005 0 001 05 1440 0001
CR1003 9056 00017 00006 0 001 05 1440 0001
CR1005 9057 00024 00009 0 001 05 1440 0001
CR1101 9058 00021 00007 0 001 05 1440 0001
CR1102 9059 00023 00009 0 001 05 1440 0001
CR1103 9060 00048 00017 0 001 05 1440 0001
CR1104 9088 00042 00015 0 001 05 1440 0001
CR1105 9117 00035 00012 0 001 05 1440 0001
CR1106 9118 00035 00013 0 001 05 1440 0001
CR1201 9064 00020 00007 0 001 05 1440 0001
CR1203 9065 00007 00002 0 001 05 1440 0001
CR1204 9066 00007 00002 0 001 05 1440 0001
CR1206 9067 00028 00010 0 001 05 1440 0001
CR1207 9068 00028 00010 0 001 05 1440 0001
CR1209 9069 00004 00002 0 001 05 1440 0001
CR1210 9070 00004 00002 0 001 05 1440 0001
CR1401 9035 00054 00019 0 001 05 1440 0001
CR1406 9089 00018 00006 0 001 05 1440 0001
CR1408 9090 00004 00002 0 001 05 1440 0001
CR1410 9091 00004 00002 0 001 05 1440 0001
CR1411 9092 00018 00006 0 001 05 1440 0001
CR1413 9093 00004 00002 0 001 05 1440 0001
CR1414 9095 00012 00005 0 001 05 1440 0001
CR1416 9096 00004 00002 0 001 05 1440 0001
CR1417 9097 00004 00002 0 001 05 1440 0001
CR1418 9099 00026 00009 0 001 05 1440 0001
CR1420 9100 00018 00006 0 001 05 1440 0001
CR1422 9101 00004 00002 0 001 05 1440 0001
CR1424 9102 00004 00002 0 001 05 1440 0001
CR1425 9103 00004 00002 0 001 05 1440 0001
CR1426 9104 00004 00002 0 001 05 1440 0001
CR1427 9106 00004 00002 0 001 05 1440 0001
CR1428 9107 00004 00002 0 001 05 1440 0001
CR1429 9108 00004 00002 0 001 05 1440 0001
CR1430 9109 00004 00002 0 001 05 1440 0001
CR1431 9110 00004 00002 0 001 05 1440 0001
CR1432 9112 00004 00002 0 001 05 1440 0001
CR1434 9113 00004 00002 0 001 05 1440 0001
CR1436 9114 00004 00002 0 001 05 1440 0001
CR1438 9115 00004 00002 0 001 05 1440 0001
CR1440 9116 00004 00002 0 001 05 1440 0001
CR1501 9999 00049 00012 0 000 00 0000 0000 BATCH TASK
CR1503 9999 00043 00016 0 000 00 0000 0000 BATCH TASK
CR1504 9999 00030 00011 0 000 00 0000 0000 BATCH TASK
CR1506 9999 00029 00011 0 000 00 0000 0000 BATCH TASK
CR1507 9999 00029 00010 0 000 00 0000 0000 BATCH TASK
CR1508 9999 00045 00016 0 000 00 0000 0000 BATCH TASK
CR1509 9999 00033 00012 0 000 00 0000 0000 BATCH TASK
```

```
CR1513 9999 00032 00012 0 000 00 0000 0000 BATCH TASK
CR1601 9999 00051 00018 0 000 00 0000 0000 BATCH TASK
CR1602 9999 00032 00011 0 000 00 0000 0000 BATCH TASK
DUMMY  9999 00000000000000000000000000000000 DUMMY TASK

CR2201 9001 00000000000000000000000000000000 MENUE
CR2203 9002 00000000000000000000000000000000 MENUE
CR2205 9074 00000000000000000000000000000000 MENUE
CR2207 9021 00000000000000000000000000000000 MENUE
ZZZZZZ
```

Appendix (D)

FILES ACCESSED BY SAMIS ONLINE SYSTEM TASKS

A file has been constructed to contain information about the different files that are accessed by all of the tasks in the functions of the SAMIS ONLINE System.

The format, description, and the contents of the file follows.

```
* File     : FACS2.SCR
* Created  : 30-AUG-1986, BY : Khaled W. Al-Dhaher
* Source   : Final Detailed Design Document for
*            the On Line System.
*            Volume 1 : Software System Specification
*            C 45 .FDD V1 C1
* Version  : 6
* Updated  : 16 SEPTEMBER 1986
*            27 NOVEMBER  1986
*             3 DECEMBER  1986
* Content : Files accessed by each function in the
* Civil registration element. As a finer granularity,
* each step-code i  nside the function is specified
* with whatever file  s it accesses, if any. Some
* step-codes are just displaying PIN's (screens),some
* other step codes access or update database records
* (files), we are no  t going to diffrentiate between
* an update and an   access to a database file,since
* both are considered as an access, to read or to write.
* In some steps in the functions, such a statement
* is encountered:
*
*  ^Initialize HEAD-OF-HOUSE-HOLD-INFORMATION record;'
*
*  This record is not a database record of course,
*  we are going to consider it as an AF record.
*
* Important note : It is assumed that the average
* number of dependants is '4'. So whenever 1/DEP is
* mentioned a '4' is   assumed and added to the average.
* If it was seen that the number is not actual, then it
* would be easy just to change the average for the step
* wherever it is mentioned that the average number of
* dependants are added.
*
* The format of the file is as follows :
*
* col 1 : ^*^         : is a comment line
*         any number : is  a new step code
*         blank      : continuing the description of
*                      the last step code.
*         other      : error
*
* A comment line :
*  any comments could be included.
*
* the line of a new step code :
*  step code,blank,comments.
*
* lines describing the files accessed :
*  blank, file accessed, blank, average # of times
*  accessed, blank, comments
*
* ------------------------------------------------------
```

```
* ELEMENT  09-CIVIL REGISTRATION ELEMENT
* ---------------------------------------
* FUNCTION 01-CITIZENSHIP REGAINING
* --------------------------------
9021
 AF 1 (activity file)
9022
 NAMEX 1  (if)
 AF 1 * fetch db rec (fetches database record)
9023
 AF 1 * fetch db rec
9003
 AF 1
9004
 AF 5 (One per dep + HOH)
 MCHECK 5 * Miscreant check (1/dep+hoh)
9027
9028
9029
9030
9031
9032
 AF 5 (1/dep+hoh)
 NAMEX 1
* Micrographics assignment MG1000
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
 SAMORD 1
 CIVNBR 1
9033
* --------------------------
* FUNCTION 02-NATURALIZATION
* --------------------------
9074
 AF 1
9075
 ALIEN 1
 AF 1
9003
 AF 1
9077
 AF 1
9007
 AF 1
9076
9004
 AF 5 (1/DEP+HOH)
9027
9028
9029
9030
9031
9083
```

```
 AF 5 (1/DEP+HOH)
 NAMEX 1 (1/DEP+HOH)
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
 SAMORD 1
 CIVNBR 1
 ALIEN  1
 LQUOTA 1
9084
* ------------------------
* FUNCTION 03-REGISTRATION
* ------------------------
9001
 AF 1
9085
 AF 1
9086
 AF 10 (1+2/DEP+HOH)
 NAMEX 1
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
 CIVNBR 1
 MCHECK 5 * miscreant check (1/DEP+HOH)
9120
 MCHECK 1

9121
 AF 1
**** 9008 might seem to be repeating 9086, actually
**** they are similar, but 9086 is done if registration
**** is done for a HOH,(accompanied with dependants,
**** which explains the n/dep access in 9086), 9008 is
**** done if the r    egistration is done for dependants.
9008
 AF 5 (HOH+1/DEP)
 NAMEX 1
* processes micrographics MG1000
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
 CIVNBR 1
 MCHECK 4 * miscreant check (1/DEP)
9009
* ---------------------------------------
* FUNCTION 04-ISSUE/ENCODE NATIONAL ID CARD
* ---------------------------------------
9001
9010
9011
 CVSTAT 1 * Update report statistical counters
* Update micrographics assignment records
 CITZEN 1
9012
```

```
9013
9014
9015
9005
* ----------------------
* FUNCTION 05-FAMILY BOOK
* ----------------------
9001
9036
9037
9038
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
* process micrographics MG1000
 MCHECK 1 * Process miscreant check NUMBCK
9009
* ----------------------------
* FUNCTION 06-REGISTER MARRIAGE
* ----------------------------
9001
9040
9041
9042
9044 (ACTIVITY 1 processing )
 CITZEN 2 (HUS+WIF)
 CIVHOH 2 (HUS+WIF)
 CVSTAT 1
9009
9045
9043 (activity 2)
 CITZEN 6 (HUS+WIF+1/DEP)
 CIVHOH 2 (HUS+WIF)
9009 (repeated, just follow the sequence)
9042 again (activity 3)
9043
 CITZEN 6 (HUS+WIF+1/DEP)
 CIVHOH 2 (HUS+WIF)
9009 (repeated, just follow the sequence)
* ----------------------------
* FUNCTION 07-REGISTER DIVORCE
* ----------------------------
9001
9061
9046
 MCHECK 1 * perform miscreant check (0,1,or 2 times)
9047
9048
9049
9050
9122
9123
 CITZEN 5 (HUS+WIF+PARENT if WIF linked
*                 +DEP if they will transfere)
```

```
 CIVHOH 2 (HUS+WIF if she is HOH
*         (or) PARENT if she is linked to him)
 CVSTAT 1 * update statistical counters
* processes micrographics assignment MG1000
9051
9124
 CITZEN 5 (HUS+DEPs)
 CIVHOH 2 (HUS+newHOH)
 CVSTAT 1 * update statistical counters
* processes micrographics assignment MG1000
9009
* -------------------------
* FUNCTION 08-REGISTER DEATH
* -------------------------
9001
9052
9009
 CVSTAT 1
* processes micrographics assignment MG1000
9053
 CITZEN 1
 CIVHOH 1
9054
 CITZEN 1
* ---------------------------------
* FUNCTION 09-WITHDRAW NATIONALITY
* ---------------------------------
9074
9061
 CVSTAT 1
 SAMORD 1
9062
 CITZEN 1
9063
* ---------------------------------
* FUNCTION 10-TRANSFER REGISTRATION
* ---------------------------------
9001
9055
 MCHECK 1 * process miscreant check
9056
9057
 CITZEN 1
 CIVHOH 1
 CVSTAT 1
* request micrographics assignment
9009
* ----------------------
* FUNCTION 11-UPDATE DATA
* ----------------------
9001
9058
9088
 CITZEN 2 (1,2,or 3 times)
```

```
 CIVHOH 2 (1,2,or 3 times)
 SPONSR 1 (0,or 1 time)
 NAMEX  1 (0,1,or 2 times)
 CVSTAT 1  * update appropriate statistical counters
9118 (note: old manual 9118, new manual 9018)
 CITZEN 2 (1,2,or 3 times)
 CIVHOH 2 (1,2,or 3 times)
 SPONSR 1 (0,or 1 time)
 NAMEX  1 (0,1,or 2 times)
 CVSTAT 1 * update appropriate statistical counters
9059
 CITZEN 1
 CVSTAT 1 * update appropriate statistical counters
* processes micrographics assignment MG1000
9060
 CITZEN 2 (1,2,or 3 times)
 CIVHOH 2 (1,2,or 3 times)
 SPONSR 1 (0,or 1 time)
 NAMEX  1 (0,1,or 2 times)
 CVSTAT 1 * update appropriate statistical counters
9117
 CITZEN 2 (1,2,or 3 times)
 CIVHOH 2 (1,2,or 3 times)
 SPONSR 1 (0,or 1 time)
 NAMEX  1 (0,1,or 2 times)
 CVSTAT 1 * update appropriate statistical counters
9009
* ----------------------
* FUNCTION 12-CHANGE NAME
* ----------------------
9002
9064
 MCHECK 1 * miscreant check (0,or 1 time)
9065
9066
9067
 NAMEX 1
 SAMORD 1
* processes micrographics assignment MG1000
9068
9069 (shown in old, but not in new manual)
9070
* ----------------------
* FUNCTION 13-CHANGE DOB
* ----------------------
9002
9064
 MCHECK 1 * miscreant check (0,or 1 time)
9065
9066
9067
 NAMEX 1
 SAMORD 1
* processes micrographics assignment MG1000
```

```
9068
9069 (shown in old, but not in new manual)
9070
* -----------------
* FUNCTION 14-QUERY
* -----------------
9001
9002
9021
9074
9035
9089
9090
9091
9092
9093
9095
9097
9099
9100
9101
9102
9103
9104
*9105 NOT IN THE MAIN MODULE LIST.
9107
9108
9109
9110
9112
9113
9114
9115
9116
X
```

Appendix (E)

TIMING DIAGRAM FOR THE MODEL

The timing diagrams have been provided to show the possible sequence of transaction states, while being processed in the model, simulating the actual SAMIS computer system transactions.

The sample timing diagram is by no means complete, or comprehensive. It just gives an example of the states, and phases the transaction will possibly face while in the model. The following is an explanation of the meaning of the indicated times on the timing diagram.

(T1) : Interaction begins. Transaction arrives to the system (FCT). Looks for a driving task (SCT).
(T2) : Delay time to find first task of the function to be executed Transaction arrives to the system (FCT). Looks for a driving task (SCT).
(T3) : No driving task (SCT). Wait until available.
(T4) : Task (SCT) becomes available. FCT will be routed to the FCT pool. SCT task will take over.
(T5) : Scheduling task for CPU.
(T6) : CPU execute cycle. This loop of schedule and execute could be repeated as much as the task requires, depending on the task CPU requirments, and the assigned CPU quantum.
(T7) : Drum I/O cycle. This cycle could be repeated as much as required by the task. One or more drum I/O operations are usually done for obtaining the index of a record in an indexed file.
(T8) : Disk I/O cycle. The drum/disk I/O cycle could be repeated as much as the task requires to access different files.
(T9) : Terminal I/O.
(T10) : FCT task takes over, and SCT is routed to SCT pool.
(T11) : FCT cheks next task to be executed in sequence of function execution. If no next task interaction is over. If there is a next task to be excuted, then check if there is a carrying SCT task. Loop back to operation indicated by time (T3).
(T12) : Delay time for (FCT) to release from system.
(T13) : End of interaction. Control task (FCT) will sink.

This cycle is done for all incoming transactions. Transactions will be executed and routed concurrently in the system. Every transaction will have its own set of data structures to keep track of its status. This serves as an example of a possible timing diagram, some special cases have not been considered for the sake of bravity.

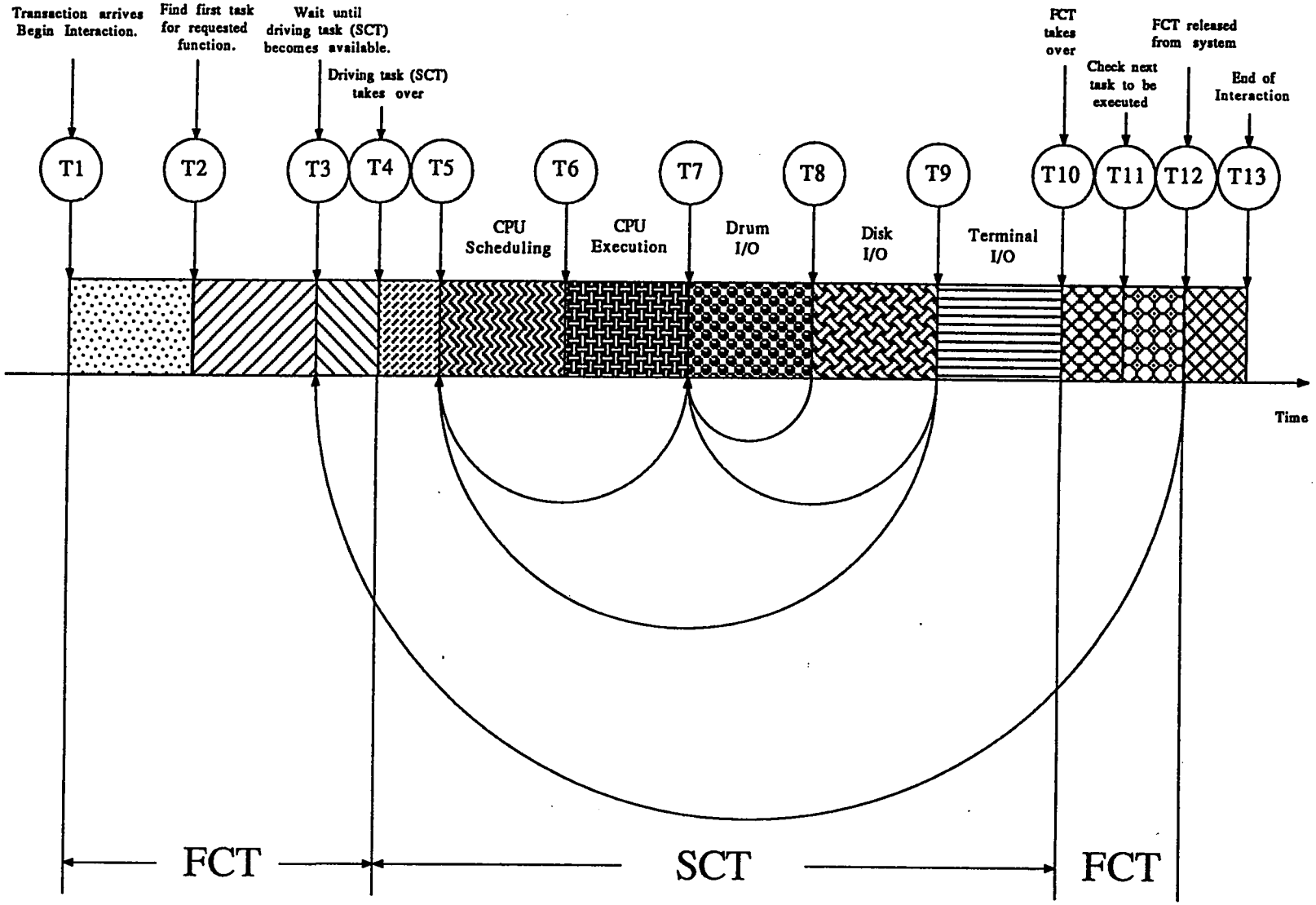


Figure E.1. Example Timing Diagram

Appendix (F)

Extraction of I-Bank and D-Bank Memory Requirements

In this appendix the method used to estimate the I-Bank and D-Bank memory requirement for different tasks in the Civil Registration Subsystem of the SAMIS Online System, will be explained.

The information provided in Fig. I.1 about the requirement of the tasks in the Civil Registration Element shows the need for combined I and D bank memory requirements. Fig. I.2 shows, among other things, the I-Bank and (x)*D-Bank requirement of a sample of tasks that were active at the time the snapshot of the system was taken, where (x) is a multiplie of D-Bank size used by the task copies active at the snapshot time.

Combining this information it is possible to establish the information provided in Figure I.3 showing the separate I and D bank memory requirement for the tasks that are in the snapshot. The approximate ratio of I to D bank memory requirement was calculated as shown in Fig. I.3. This ratio could now be used to estimate the separate I and D bank requirements of all tasks that are in the Civil Registration Element provided in Fig. I.1.

This ratio was used to produce the estimated I and D bank requirements that are listed in Appendix (C).

As an enhancement of the model, the inclusion of actual I and D bank requirement of different tasks in the data structures could be done.

```
CM3040-XQT/IMPXQT SIZE:6
CM2050-XQT/IMPXQT SIZE:30
CM2060-XQT/IMPXQT SIZE:8
CM3301-XQT/IMPXQT SIZE:85
CM3302-XQT/IMPXQT SIZE:46
CM3303-XQT/IMPXQT SIZE:77
CM3305-XQT/IMPXQT SIZE:84
CM3309-XQT/IMPXQT SIZE:78
CM3315-XQT/IMPXQT SIZE:65
CM3316-XQT/IMPXQT SIZE:77
CM3319-XQT/IMPXQT SIZE:40
CM3320-XQT/IMPXQT SIZE:21
CM3321-XQT/IMPXQT SIZE:48
CM3321-XQT/IMPXQT SIZE:73
CM5710-XQT/IMPXQT SIZE:21
CM6010-XQT/IMPXQT SIZE:12
CM7030-XQT/IMPXQT SIZE:32
CM7120-XQT/IMPXQT SIZE:43
CM7210-XQT/IMPXQT SIZE:31
CM7310-XQT/IMPXQT SIZE:37
CM7410-XQT/IMPXQT SIZE:42
CM7510-XQT/IMPXQT SIZE:36
CM7610-XQT/IMPXQT SIZE:27
CM9420-XQT/IMPXQT SIZE:42
CM9510-XQT/IMPXQT SIZE:6
CM9520-XQT/IMPXQT SIZE:34
CM9710-XQT/IMPXQT SIZE:26
CM9810-XQT/IMPXQT SIZE:26
CM9910-XQT/IMPXQT SIZE:31
CNTLOD-XQT/IMPXQT SIZE:14
CR0115-XQT/IMPXQT SIZE:37
CR0116-XQT/IMPXQT SIZE:24
CR0117-XQT/IMPXQT SIZE:6
CR0118-XQT/IMPXQT SIZE:6
CR0119-XQT/IMPXQT SIZE:31
CR0120-XQT/IMPXQT SIZE:26
CR0121-XQT/IMPXQT SIZE:9
CR0122-XQT/IMPXQT SIZE:50
CR0123-XQT/IMPXQT SIZE:6
CR0124-XQT/IMPXQT SIZE:6
CR0125-XQT/IMPXQT SIZE:6
CR0126-XQT/IMPXQT SIZE:6
CR0127-XQT/IMPXQT SIZE:6
CR0128-XQT/IMPXQT SIZE:6
CR0129-XQT/IMPXQT SIZE:6
CR0130-XQT/IMPXQT SIZE:41
CR0131-XQT/IMPXQT SIZE:6
CR0201-XQT/IMPXQT SIZE:19
CR0203-XQT/IMPXQT SIZE:36
CR0205-XQT/IMPXQT SIZE:20
CR0207-XQT/IMPXQT SIZE:45
CR0214-XQT/IMPXQT SIZE:6
CR0301-XQT/IMPXQT SIZE:31
CR0302-XQT/IMPXQT SIZE:59
CR0304-XQT/IMPXQT SIZE:25
CR0306-XQT/IMPXQT SIZE:9
CR0308-XQT/IMPXQT SIZE:60
CR0310-XQT/IMPXQT SIZE:6
CR0312-XQT/IMPXQT SIZE:6
CR0430-XQT/IMPXQT SIZE:25
CR0432-XQT/IMPXQT SIZE:20
CR0434-XQT/IMPXQT SIZE:6
CR0435-XQT/IMPXQT SIZE:23
CR0445-XQT/IMPXQT SIZE:6
CR0501-XQT/IMPXQT SIZE:28
CR0503-XQT/IMPXQT SIZE:21
CR0505-XQT/IMPXQT SIZE:34
CR0601-XQT/IMPXQT SIZE:31
CR0603-XQT/IMPXQT SIZE:6
CR0605-XQT/IMPXQT SIZE:9
CR0607-XQT/IMPXQT SIZE:32
CR0609-XQT/IMPXQT SIZE:22
CR0611-XQT/IMPXQT SIZE:9
CR0701-XQT/IMPXQT SIZE:26
CR0702-XQT/IMPXQT SIZE:24
CR0703-XQT/IMPXQT SIZE:9
CR0721-XQT/IMPXQT SIZE:22
CR0705-XQT/IMPXQT SIZE:9
CR0706-XQT/IMPXQT SIZE:22
CR0707-XQT/IMPXQT SIZE:10
CR0708-XQT/IMPXQT SIZE:34
CR0709-XQT/IMPXQT SIZE:33
CR0801-XQT/IMPXQT SIZE:24
CR0803-XQT/IMPXQT SIZE:23
CR0805-XQT/IMPXQT SIZE:21
CR0901-XQT/IMPXQT SIZE:35
CR0903-XQT/IMPXQT SIZE:31
CR0905-XQT/IMPXQT SIZE:6
CR1001-XQT/IMPXQT SIZE:20
CR1003-XQT/IMPXQT SIZE:23
CR1005-XQT/IMPXQT SIZE:33
CR1101-XQT/IMPXQT SIZE:28
CR1102-XQT/IMPXQT SIZE:32
CR1103-XQT/IMPXQT SIZE:65
CR1104-XQT/IMPXQT SIZE:57
CR1105-XQT/IMPXQT SIZE:47
CR1106-XQT/IMPXQT SIZE:48
CR1201-XQT/IMPXQT SIZE:27
CR1203-XQT/IMPXQT SIZE:9
CR1204-XQT/IMPXQT SIZE:9
CR1206-XQT/IMPXQT SIZE:38
CR1207-XQT/IMPXQT SIZE:38
CR1209-XQT/IMPXQT SIZE:6
CR1210-XQT/IMPXQT SIZE:6
CR1401-XQT/IMPXQT SIZE:73
CR1406-XQT/IMPXQT SIZE:24
CR1408-XQT/IMPXQT SIZE:6
CR1410-XQT/IMPXQT SIZE:6
CR1411-XQT/IMPXQT SIZE:24
CR1413-XQT/IMPXQT SIZE:6
CR1414-XQT/IMPXQT SIZE:17
CR1416-XQT/IMPXQT SIZE:6
CR1417-XQT/IMPXQT SIZE:6
CR1418-XQT/IMPXQT SIZE:35
CR1420-XQT/IMPXQT SIZE:24
CR1422-XQT/IMPXQT SIZE:6
CR1424-XQT/IMPXQT SIZE:6
CR1425-XQT/IMPXQT SIZE:6
CR1426-XQT/IMPXQT SIZE:6
CR1427-XQT/IMPXQT SIZE:6
CR1428-XQT/IMPXQT SIZE:6
CR1429-XQT/IMPXQT SIZE:6
CR1430-XQT/IMPXQT SIZE:6
CR1431-XQT/IMPXQT SIZE:6
CR1432-XQT/IMPXQT SIZE:6
CR1434-XQT/IMPXQT SIZE:6
CR1501-XQT/IMPXQT SIZE:61
CR1503-XQT/IMPXQT SIZE:59
CR1504-XQT/IMPXQT SIZE:41
CR1506-XQT/IMPXQT SIZE:40
CR1507-XQT/IMPXQT SIZE:39
CR1508-XQT/IMPXQT SIZE:61
CR1509-XQT/IMPXQT SIZE:45
CR1513-XQT/IMPXQT SIZE:44
CR1601-XQT/IMPXQT SIZE:69
CR1602-XQT/IMPXQT SIZE:42
CR2201-XQT/IMPXQT SIZE:7
CR2203-XQT/IMPXQT SIZE:6
CR2205-XQT/IMPXQT SIZE:7
CR2207-XQT/IMPXQT SIZE:7
CX0332-XQT/IMPXQT SIZE:24
DATES-XQT/IMPXQT SIZE:14
DBCOLE-XQT/IMPXQT SIZE:9
DBNCRT-XQT/IMPXQT SIZE:5
DBPART-XQT/IMPXQT SIZE:10
DBSORC-XQT/IMPXQT SIZE:9
DL0001-XQT/IMPXQT SIZE:7
DL0101-XQT/IMPXQT SIZE:25
DL0104-XQT/IMPXQT SIZE:35
DL0111-XQT/IMPXQT SIZE:6
DL0115-XQT/IMPXQT SIZE:32
DL0201-XQT/IMPXQT SIZE:22
DL0204-XQT/IMPXQT SIZE:41
DL0208-XQT/IMPXQT SIZE:6
DL0301-XQT/IMPXQT SIZE:16
DL0303-XQT/IMPXQT SIZE:15
DL0305-XQT/IMPXQT SIZE:18
DL0401-XQT/IMPXQT SIZE:25
DL0405-XQT/IMPXQT SIZE:34
DL0501-XQT/IMPXQT SIZE:24
DL0505-XQT/IMPXQT SIZE:20
DL0507-XQT/IMPXQT SIZE:6
DL0601-XQT/IMPXQT SIZE:21
DL0605-XQT/IMPXQT SIZE:21
DL0607-XQT/IMPXQT SIZE:6
DL0701-XQT/IMPXQT SIZE:49
DL0801-XQT/IMPXQT SIZE:22
DL0803-XQT/IMPXQT SIZE:75
DL0804-XQT/IMPXQT SIZE:84
DL0805-XQT/IMPXQT SIZE:41
DL0806-XQT/IMPXQT SIZE:61
DL0807-XQT/IMPXQT SIZE:41
DL0809-XQT/IMPXQT SIZE:69
DL0808-XQT/IMPXQT SIZE:62
DL0810-XQT/IMPXQT SIZE:64
DL0811-XQT/IMPXQT SIZE:44
DL0812-XQT/IMPXQT SIZE:41
DL0813-XQT/IMPXQT SIZE:68
DL0814-XQT/IMPXQT SIZE:59
DL0815-XQT/IMPXQT SIZE:42
DL0816-XQT/IMPXQT SIZE:49
DL0817-XQT/IMPXQT SIZE:50
DL0818-XQT/IMPXQT SIZE:60
DL0819-XQT/IMPXQT SIZE:60
DL0901-XQT/IMPXQT SIZE:17
```

Figure F.1. Supplied List of Task Memory Requirement

```
   S0021STST015  UNGER        08/06/85     15:30:22     0334      011-047
! DISPL,B
CENTER: SO      DATE: 08/06/85      TIME: 15:30:45      DISPLAY OF  37 TASKS

***  37 BATCH TASKS                    PAGES=512 WORDS
TASK USERID   XQT      SIZE    THRU  IOI    ELAPS        CPU   SWAP/T   IO/T TDISK  P
                        I  D
022*-PSL300 ERRTRM 30- 40    1307   280 32:51:04*      .90      .00    .00     0  0
025*-ONL001 VALIDT 33- 43      40   428 32:50:54*    29.21     3.33    .12     0  0
027*-ONL001 OPAUTH 38- 48      14   403 32:50:23*    81.92     3.85    .61     0  0
030*-QNL001 PC0860  5+  7      58   486 32:50:10*    20.35     1.50    .07     0  0
032*-ONL001 BC5025  3+  6     167   477 32:49:51*     7.05      .53    .03     0  0
056*-ONL001 BC5006 21+ 16       9   355 32:44:45*   128.58     2.05    .71     0  0
120*-ONL001 PC0850  3+  6     323   472 32:16:37*     3.59      .31    .01     0  0
126*-ONL001 PC0809 25+ 15      99   388 32:12:39*    11.65      .14    .06     0  0
140*-ONL001 PC0801 52+ 30       0   357  6:38:42*   324.37    22.93   7.12     0  0
155*-ONL001 CR0301 21+ 15      19   428 31:53:13*    58.21     2.72    .17     0  0
167*-ONL001 ERRTRM 30- 39      19   418 25:27:58*    47.34     1.81    .41     0  0
207 -ONL001 PC0807 13+  9      44   408  6:29:02*     5.22     1.43    .11     0  0
217 -ONL001 PC0808 27+ 15      29   370  6:27:21*     7.77     1.52    .19     0  0
270  PTS001 PTS    43- 24      67   280  0:50:23*      .45      .09    .07     0  1
273 -ONL100 ERRTRM 31- 39     151   288  3:43:38*      .88      .10    .04     0  0
275 -ONL201 ERRTRM 31- 39      69   339  3:42:33*     1.91      .39    .11     0  0
302 -ONL001 PC0848 33+ 20      18   429  6:09:40*    12.10     2.90    .18     0  0
303 -ONL001 PC0849  3+  6      85   462  6:07:57*     2.59     1.16    .06     0  0
304*-ONL001 PC0801 52+ 30       1   366  0:44:06*    15.48    13.57   3.09     0  0
305*-ONL001 PC0801 52+ 30       1   366  0:43:50*    20.31    18.02   4.32     0  0
325 -ONL001 MC0100 10+  9      20   303  0:18:40*      .53     1.62    .30     0  0
326 -ONL001 MC0200 17+ 12      19   305  0:17:29*      .54     2.46    .29     0  0
327 -ONL001 AC0003  3+  6      15   237  0:12:16*      .48      .52    .29     0  0
333*-ONL001 MS0107  8+ 10     631   426 29:09:45*     1.66      .12    .01     0  0
376*-ONL001 BC5005 37+ 27       6   382 23:55:32*   136.62     3.60    .46     0  0
407*-ONL001 PC0801 52+ 31       0   359  9:17:08*   350.29    18.48   5.58     0  0
416* TST018 MENTOR 48- 37       2   344  8:53:02*   113.92      .54   1.30    16 30
417  TSE010 MESAGE 27- 22       9   150  3:07:56*    12.44      .43    .09     2 26
434  TST018 PAUSE  11-  7      10   329  8:30:40*    28.82     1.15    .62    17 30
463* TSO110 FAMGMC 16- 18     115   115  8:03:22*     2.50      .11    .07     3 30
563 -ONL001 BC5017 12+ 10     156   245  2:37:46*      .61      .23    .03     0  0
727  TSE010 341TST 26- 68      10   146  7:10:03*    23.92      .56    .01     2 26
747 -ONL001 CR0302 47+ 35       5   368  7:05:07*    43.34     4.07    .82     0  0
751 -ONL001 AC0101 22+ 19      22   361  2:02:18*     3.23     2.68    .28     0  0
766 -ONL001 AC0102  7+ 40       7   394  1:59:08*     9.95     2.01    .11     0  0
767 -ONL001 AC0103 24+ 22       5   354  1:58:51*    12.35     3.17   1.26     0  0
774 -ONL001 AC0104  3+  6      58   428  1:57:24*     1.20     1.08    .08     0  0

***    1 MTS TERMINALS CONNECTED
```

Figure F.2. Supplied Snapshot of Different Task Information

Task	(I+D)-Size	(I)-Size	(D)-Size
PC0860	8	5	3
BC5025	6	3	3
BC5006	29	12	17
PC0850	6	3	3
PC0809	32	25	7
PC0801	62	52	10
CR0301	31	21	10
PC0807	18	13	5
PC0808	34	27	7
PC0848	42	33	9
PC0849	6	3	3
MC0100	15	10	5
MC0200	23	17	6
AC0003	6	3	3
BC5005	47	37	10
PC0801	62	52	10
BC5017	18	12	6
CR0302	59	47	12
AC0101	31	22	9
AC0102	12	7	5
AC0103	33	24	9
AC0104	7	3	4
SUM :	587	431	156
Number :	22	22	22
Average :	26.682	19.590	7.091
Ratio : (SUM/587)	1.000	0.734	0.266

Figure F.3. Calculated Weighted Average of I and D Bank Requirement for Supplied Tasks